

普通高等教育电气信息类规划教材



免费电子教案下载

www.cmpedu.com



电气控制与西门子 S7-300 PLC 编程技术

刘忠超 田金云 主 编

刘增磊 杨 旭 李名莉 副主编





机械工业出版社
CHINA MACHINE PRESS

VISIT…

LANZAROTE Caliente.COM

普通高等教育电气信息类规划教材

电气控制与西门子 S7－300 PLC 编程技术

刘忠超　田金云　主编
刘增磊　杨　旭　李名莉　副主编
肖东岳　范伟强　参编
盖晓华　主审



机　械　工　业　出　版　社

西门子 S7 系列 PLC 性能卓越，广泛应用于各种工业生产中，在工控市场占有非常大的份额。

本书内容包括电气控制与 PLC 编程技术两部分。电气控制部分包括常用低压电器的工作原理及选型、电气控制电路的分析与设计等；PLC 编程技术以西门子 S7 - 300 PLC 及其编程软件 STEP 7 为主线，重点介绍了 PLC 工作原理、硬件结构、编程软件、指令系统、程序结构以及 PLC 控制工程实例等。本书内容丰富、全面系统、实用性强，理论结合实例清晰易懂，可以让读者快速掌握西门子自动化的相关技术。

本书可作为高等院校自动化、电气工程及其自动化、机电一体化、计算机控制及相关专业本科教材，也可作为工程技术人员培训及自学参考书，同时对西门子自动化系统的用户也是很好的参考书。

图书在版编目(CIP)数据

电气控制与西门子 S7 - 300 PLC 编程技术 / 刘忠超，田金云主编 . —北京：机械工业出版社，2017. 9

普通高等教育电气信息类规划教材

ISBN 978-7-111-57621-1

Ⅰ. ①电…　Ⅱ. ①刘…　②田…　Ⅲ. ①电气控制 - 高等学校 - 教材 ②PLC 技术 - 高等学校 - 教材　Ⅳ. ①TM571

中国版本图书馆 CIP 数据核字(2017)第 179686 号

机械工业出版社(北京市百万庄大街 22 号　邮政编码 100037)
策划编辑：尚　晨　　责任编辑：尚　晨
责任校对：张艳霞　　责任印制：李　昂
三河市国英印务有限公司印刷

2017 年 8 月第 1 版 · 第 1 次印刷
184mm × 260mm · 18. 5 印张 · 451 千字
0001 - 3000 册
标准书号：ISBN 978-7-111-57621-1
定价：49. 80 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

电话服务
服务咨询热线：(010)88379833
读者购书热线：(010)88379649

网络服务
机 工 官 网：www. cmpbook. com
机 工 官 博：weibo. com/cmp1952
教育服务网：www. cmpedu. com
金 书 网：www. golden - book. com

封面无防伪标均为盗版

前　言

PLC 是可编程序控制器的简称，是一种在继电器接触器控制技术和计算机控制技术的基础上发展起来的新型工业自动控制装置。PLC 以其控制能力强、可靠性高、配置灵活、编程简单、使用方便、易于扩展等优点，成为工业控制领域中增长速度最迅猛的工业控制设备之一，在机械制造、石油化工、冶金钢铁、汽车、轻工业等领域得到了广泛的应用。

本书以德国西门子公司的 S7－300 PLC 为主，全面介绍了 PLC 的基础知识和应用设计方法，全书共分为 10 章。第 1 章介绍了电气控制的相关知识；第 2 章主要介绍可编程控制器的发展、定义和工作原理等；第 3 章介绍了 S7－300 PLC 的硬件体系和模块特性；第 4 章主要介绍了 S7－300 PLC 的编程软件 STEP 7 的使用方法；第 5 章介绍了 S7－300 PLC 的编程基础；第 6 章重点讲解了 S7－300 PLC 的指令系统；第 7 章介绍了 PLC 编程方法，重点是结构化编程；第 8 章介绍了数据块和组织块的使用方法；第 9 章介绍了模拟量控制的处理及闭环控制；第 10 章结合工程实例给出了 PLC 控制系统的设计原则、内容和步骤。

本书由南阳理工学院刘忠超、田金云任主编，南阳理工学院刘增磊、杨旭、河南工业职业技术学院李名莉任副主编。第 1 章由刘增磊编写，第 3 章、第 4 章由杨旭编写，第 2 章、第 8 章由李名莉编写，第 6 章由刘忠超编写，第 7 章、第 9 章、第 10 章由田金云编写，第 5 章由中国矿业大学（北京）机电与信息工程学院范伟强编写。肖东岳对全书进行了校对，朱清慧、翟天嵩、盖晓华、崔世林、殷华文参加了本书的出版和审阅工作，在此一并表示衷心的感谢。刘忠超负责本书的结构和组织安排，并对全书进行了整理和统稿。

由于编者水平和时间有限，书中难免有疏漏之处，恳请广大读者批评指正。

编者

目　录

前言
第 1 章　电气控制基础 …… 1
1.1　常用低压电器 …… 1
1.1.1　刀开关 …… 2
1.1.2　熔断器 …… 4
1.1.3　断路器 …… 6
1.1.4　接触器 …… 8
1.1.5　控制继电器 …… 9
1.1.6　主令电器 …… 16
1.2　电气图中的图形符号和文字符号 …… 20
1.2.1　电器的文字符号 …… 20
1.2.2　电器的图形符号 …… 20
1.3　电气图基本知识 …… 25
1.4　三相异步电动机的基本控制电路 …… 27
1.4.1　基本控制环节 …… 27
1.4.2　三相异步电动机起动控制 …… 30
1.4.3　三相异步电动机制动控制 …… 32
1.4.4　三相异步电机调速控制线路 …… 33
习题 …… 35
第 2 章　可编程序控制器概述 …… 36
2.1　PLC 简介 …… 36
2.1.1　PLC 的产生 …… 36
2.1.2　PLC 的发展 …… 37
2.1.3　PLC 的定义 …… 37
2.1.4　PLC 的分类 …… 38
2.1.5　PLC 的特点 …… 39
2.1.6　PLC 性能指标 …… 41
2.2　PLC 硬件组成 …… 41
2.3　PLC 编程环境和工作原理 …… 44
2.3.1　PLC 编程环境 …… 44
2.3.2　PLC 编程语言及编程软件 …… 45
2.3.3　PLC 的工作原理 …… 47

2.4 PLC 产品概况及发展趋势 …… 48
2.5 PLC 在工业自动化中的地位和作用 …… 50
2.6 SIMATIC 自动化控制系统 …… 50
2.6.1 SIMATIC 的组成 …… 50
2.6.2 全集成自动化 …… 53
习题 …… 53
第 3 章 S7-300 PLC 硬件系统 …… 54
3.1 S7-300 PLC 硬件简介 …… 54
3.2 S7-300 PLC 机架 …… 54
3.3 S7-300 PLC 电源模块 …… 55
3.4 S7-300 PLC CPU 模块 …… 56
3.4.1 CPU 模块的分类 …… 56
3.4.2 CPU 面板介绍 …… 57
3.4.3 CPU 模块存储器 …… 60
3.5 S7-300 PLC 接口模块 …… 61
3.6 S7-300 PLC 信号模块 …… 62
3.6.1 数字量模块 …… 62
3.6.2 模拟量模块 …… 65
3.6.3 数字量仿真模块 …… 67
3.7 其他模块 …… 68
3.8 编程设备 PG/PC …… 69
3.9 人机操作界面 …… 70
3.9.1 人机界面的概念 …… 70
3.9.2 人机界面的功能 …… 70
3.9.3 WinCC 软件 …… 71
3.10 S7-300 PLC 模块的安装与扩展 …… 72
3.10.1 模块配放原则 …… 72
3.10.2 模块安装规范 …… 72
3.10.3 模块安装步骤 …… 73
3.10.4 模块扩展 …… 74
3.11 S7-300 PLC 模块编址 …… 76
3.11.1 数字量 I/O 编址 …… 76
3.11.2 模拟量 I/O 编址 …… 77
3.11.3 面向用户编址 …… 77
习题 …… 78
第 4 章 编程软件 STEP 7 的介绍与使用 …… 79
4.1 STEP 7 概述 …… 79

4.2　安装与卸载 STEP 7 …… 80
4.2.1　安装系统配置 …… 80
4.2.2　STEP 7 授权 …… 81
4.2.3　安装 STEP 7 …… 81
4.2.4　卸载 STEP 7 …… 84
4.3　SIMATIC 管理器概述 …… 85
4.4　项目的管理与创建 …… 85
4.4.1　STEP 7 创建项目步骤 …… 85
4.4.2　项目结构 …… 85
4.4.3　创建项目 …… 86
4.4.4　编辑项目 …… 88
4.5　硬件组态 …… 89
4.5.1　硬件组态步骤 …… 89
4.5.2　参数设置 …… 90
4.5.3　硬件组态目录更新 …… 92
4.6　符号表 …… 93
4.6.1　符号表的创建 …… 94
4.6.2　符号表的管理 …… 95
4.7　逻辑块的生成 …… 98
4.7.1　逻辑块的组成 …… 98
4.7.2　程序编辑器的设置 …… 98
4.7.3　输入程序的方式 …… 99
4.7.4　程序块上锁 …… 100
4.7.5　程序编辑器 …… 101
4.8　硬件接口与程序下载及上传 …… 104
4.8.1　硬件接口 …… 104
4.8.2　程序下载及上传 …… 105
4.9　程序调试工具 PLCSIM …… 106
4.9.1　S7－PLCSIM 的介绍 …… 106
4.9.2　S7－PLCSIM 的使用 …… 107
4.9.3　S7－PLCSIM 的调试举例 …… 109
4.9.4　PLCSIM 与真实 PLC 的差别 …… 110
习题 …… 111
第 5 章　S7－300 PLC 的编程基础 …… 112
5.1　IEC61131－3 国际标准简介 …… 112
5.2　S7－300 PLC 编程语言 …… 113
5.3　梯形图绘制原则 …… 115

5.4　变量与数制 …… 115
5.4.1　变量 …… 115
5.4.2　常量和数制 …… 115
5.5　数据类型 …… 117
5.5.1　基本数据类型 …… 117
5.5.2　复杂数据类型 …… 120
5.5.3　参数类型 …… 121
5.6　PLC 中的存储区与寄存器 …… 122
5.6.1　系统存储器 …… 122
5.6.2　CPU 中的寄存器 …… 123
5.7　指令结构与寻址方式 …… 125
5.7.1　指令结构 …… 125
5.7.2　寻址方式 …… 126
5.8　S7－300 PLC 程序结构 …… 129
习题 …… 132
第 6 章　S7－300 PLC 指令系统 …… 133
6.1　位逻辑指令 …… 133
6.1.1　位逻辑指令简介 …… 133
6.1.2　位逻辑指令使用方法 …… 135
6.1.3　位操作指令 …… 138
6.1.4　位测试指令 …… 141
6.2　定时器指令 …… 145
6.2.1　定时器指令简介 …… 145
6.2.2　定时器指令分类 …… 147
6.2.3　定时器举例 …… 155
6.3　计数器指令 …… 158
6.3.1　计数器简介 …… 158
6.3.2　计数器指令分类 …… 159
6.3.3　计数器举例 …… 162
6.4　数据处理指令 …… 165
6.4.1　装入指令与传送指令 …… 165
6.4.2　比较指令 …… 167
6.4.3　数据转换指令 …… 171
6.4.4　移位指令 …… 175
6.5　运算指令 …… 183
6.5.1　算术运算指令 …… 183
6.5.2　字逻辑运算指令 …… 189

6.5.3 累加器指令 …… 190
6.6 控制指令 …… 192
6.6.1 逻辑控制指令 …… 192
6.6.2 程序控制指令 …… 196
6.6.3 主控继电器指令 …… 197
6.6.4 数据块指令 …… 199
6.7 库分类及应用 …… 200
6.7.1 库的分类 …… 200
6.7.2 库的应用 …… 201
6.7.3 库的生成 …… 202
习题 …… 208
第 7 章 结构化编程 …… 211
7.1 编程方式和程序结构 …… 211
7.1.1 编程方式 …… 211
7.1.2 程序结构 …… 211
7.2 结构化编程的实现 …… 214
7.2.1 结构化编程 …… 215
7.2.2 实现形式 …… 215
7.3 功能与功能块 …… 219
7.3.1 功能块 …… 219
7.3.2 功能 …… 223
7.3.3 功能的调用 …… 225
7.3.4 功能块的调用 …… 226
7.4 结构化程序设计实例 …… 226
7.4.1 编辑并调用有参功能（FC） …… 226
7.4.2 多重背景使用举例 …… 229
习题 …… 235
第 8 章 数据块和组织块 …… 236
8.1 数据块 …… 236
8.1.1 数据块的类型 …… 236
8.1.2 数据块的生成 …… 237
8.1.3 数据块的访问 …… 238
8.2 S7-300 PLC 的组织块 …… 239
8.2.1 组织块的分类 …… 239
8.2.2 中断 …… 240
8.2.3 启动组织块 …… 242
8.2.4 时间延时中断组织块 …… 243

8.2.5 硬件中断组织块 …… 245
8.3 定期执行组织块 …… 248
8.3.1 日期时间中断组织块 …… 248
8.3.2 循环中断组织块 …… 252
8.4 错误处理组织块 …… 255
8.4.1 错误处理概述 …… 255
8.4.2 错误的分类 …… 256
8.4.3 同步错误组织块 …… 256
8.4.4 冗余错误组织块 …… 256
习题 …… 257
第9章 S7-300/400 PLC在模拟量闭环控制中的应用 …… 259
9.1 PID控制器 …… 259
9.1.1 PID控制的基本原理 …… 259
9.1.2 PID控制器的数字化 …… 259
9.1.3 PID控制器参数与系统性能关系 …… 260
9.2 连续PID控制模块FB41 …… 261
9.2.1 设定值与过程变量的处理 …… 261
9.2.2 控制器算法 …… 262
9.2.3 输出值的处理 …… 263
9.3 脉冲发生器模块FB43 …… 263
9.3.1 脉冲发生器的功能与结构 …… 263
9.3.2 两步控制器 …… 265
9.3.3 三步控制器 …… 265
9.4 闭环控制示例 …… 267
9.4.1 系统简介 …… 267
9.4.2 硬件选型及信号类型设置 …… 267
9.4.3 程序设计 …… 268
习题 …… 270
第10章 PLC控制系统设计 …… 271
10.1 系统设计的原则和内容 …… 271
10.1.1 设计原则 …… 271
10.1.2 设计内容 …… 271
10.1.3 设计步骤 …… 272
10.2 硬件设计 …… 272
10.2.1 PLC的选型 …… 272
10.2.2 I/O模块的选型 …… 273
10.2.3 PLC容量估算 …… 274

10.3　软件设计…………………………………………………………………… 275
10.3.1　设计前准备工作 ……………………………………………………… 275
10.3.2　编写程序 ……………………………………………………………… 275
10.3.3　程序测试 ……………………………………………………………… 275
10.4　系统调试…………………………………………………………………… 276
10.4.1　调试步骤 ……………………………………………………………… 276
10.4.2　调试方法 ……………………………………………………………… 276
10.5　应用实例…………………………………………………………………… 277
10.5.1　水箱水温自动控制系统……………………………………………… 277
10.5.2　功能需求 ……………………………………………………………… 277
10.5.3　控制方案设计 ………………………………………………………… 277
10.5.4　硬件系统设计 ………………………………………………………… 277
10.5.5　程序设计 ……………………………………………………………… 279
10.5.6　上位机软件设计 ……………………………………………………… 283
10.5.7　系统调试 ……………………………………………………………… 284
习题……………………………………………………………………………… 285
参考文献……………………………………………………………………… 286

第 1 章　电气控制基础

电气控制技术是以各类电动机为动力的传动装置或者系统为对象，以实现生产过程自动化的控制技术。继电器——接触器控制系统至今仍是许多生产机械设备广泛采用的基本电气控制形式，也是学习更先进电气控制系统的基础。本章主要介绍电气控制的基本原理及基本线路。

1.1　常用低压电器

凡是能自动或手动地接通或断开电路，连续或间断地改变电路参数，以实现对电路或非电对象的切换、控制、检测、保护、变换和调节的电器元件统称为电器。简单地说，电器就是一种能控制电的工具。

电气是电能的生产、传输、分配、使用和电工装备制造等学科或工程领域的统称。它是以电能、电气设备和电气技术为手段来创造、维持与改善限定空间和环境的一门科学，涵盖电能的转换、利用和研究三方面，包括基础理论、应用技术、设施设备等。电气是广义词，既可以指一种行业，一种专业，也可指一种技术，而不具体指某种产品。

电气控制主要分为两大类：一种是传统的以继电器、接触器为主搭建起来的逻辑电路，即继电器——接触器控制；另一种是基于 PLC（Programmable Logic Controller，可编程序控制器的系统——PLC 控制）。

低压电器被广泛地应用于工业电气和建筑电气控制系统中，它是实现继电器——接触器控制的主要电器元件。通常将额定工作电压在交流 1200 V，直流 1500 V 以下，在电路中起通断、保护、控制或调节等作用的电气设备（器件）的总称为低压电器。

低压电器种类繁多，功能和构造各异，用途广泛，工作原理各不相同，常用低压电器的分类方法也很多。

（1）按用途或控制对象分类

① 配电电器：主要用于低压配电系统中。要求系统发生故障时准确动作、可靠工作，在规定条件下具有相应的动稳定性与热稳定性，使电器不会被损坏。常用的配电电器有刀开关、转换开关、熔断器、断路器等。

② 控制电器：主要用于电气传动系统中。要求寿命长、体积小、重量轻且动作迅速、准确、可靠。常用的控制电器有接触器、继电器、起动器、主令电器、电磁铁等。

（2）按动作方式分类

① 自动电器：依靠自身参数的变化或外来信号的作用，自动完成接通或分断等动作，如接触器、继电器等。

② 手动电器：用手动操作来进行切换的电器，如刀开关、转换开关、按钮等。

（3）按触点类型分类

① 有触点电器：利用触点的接通和分断来切换电路，如接触器、刀开关、按钮等。

② 无触点电器：无可分离的触点。主要利用电子元件的开关效应，即导通和截止来实现电路的通、断控制，如接近开关、霍尔开关、电子式时间继电器、固态继电器等。

（4）按工作原理分类

① 电磁式电器：根据电磁感应原理动作的电器，如接触器、继电器、电磁铁等。

② 非电量控制电器：依靠外力或非电量信号（如速度、压力、温度等）的变化而动作的电器，如转换开关、行程开关、速度继电器、压力继电器、温度继电器等。

1.1.1 刀开关

刀开关是一种手动电器，常用的刀开关有 HD 型单投刀开关、HS 型双投刀开关、HR 型熔断器式刀开关、HZ 型组合开关、HK 型闸刀开关、HY 型倒顺开关等。

HD 型单投刀开关、HS 型双投刀开关、HR 型熔断器式刀开关主要用于在成套配电装置中作为隔离开关，装有灭弧装置的刀开关也可以控制一定范围内的负载线路。作为隔离开关的刀开关的容量比较大，其额定电流在 100 A ~ 1500 A 之间，主要用于供配电线路的电源隔离作用。隔离开关没有灭弧装置，不能操作带负载的线路，只能操作空载线路或电流很小的线路，如小型空载变压器、电压互感器等。操作时应注意，停电时应将线路的负载电流用断路器、负载开关等开关电器切断后再将隔离开关断开，送电时操作顺序相反。隔离开关断开时有明显的断开点，有利于检修人员的停电检修工作。隔离刀开关由于控制负载能力很小，也没有保护线路的功能，所以通常不能单独使用，一般要和能切断负载电流和故障电流的电器（如熔断器、断路器和负载开关等电器）一起使用。

HZ 型组合开关、HK 型闸刀开关一般用于电气设备及照明线路的电源开关。HY 型倒顺开关、HH 型铁壳开关装有灭弧装置，一般可用于电气设备的起动、停止控制。

（1）HD 型单投刀开关

HD 系列单投、HS 系列双投刀开关适用于交流 50 Hz、额定电压至 380 V、直流至 440 V；额定电流至 1500 A 的成套配电装置中，作为不频繁地手动接通和分断交、直流电路或作隔离开关用。HD 型单投刀开关按极数分为 1 极、2 极、3 极、4 极四种，其实物图如图 1-1 所示。

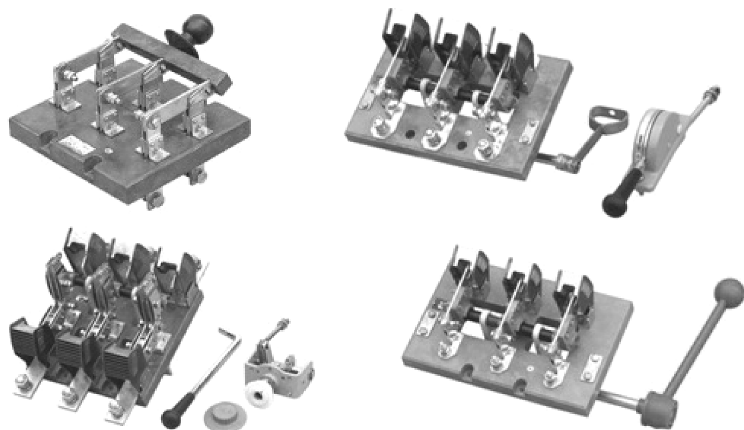


图 1-1　HD 型单投刀开关实物图

图 1-2a ~ c 为刀开关的图形符号和文字符号。其中图 1-2a 为一般图形符号，b 为手动符号，c 为三极单投刀开关符号。

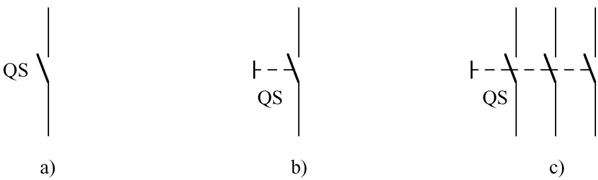


图 1-2　HD 型单投刀开关图形符号

a）一般图形符号　b）手动符号　c）三极单投刀开关符号

当刀开关用作隔离开关时，其图形符号上加有一横杠，如图 1-3a、图 1-3b、图 1-3c 所示。

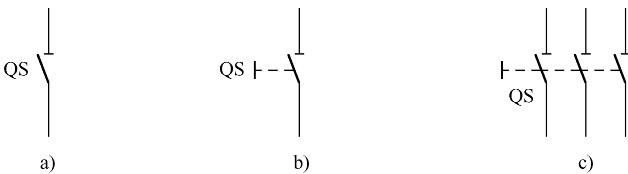
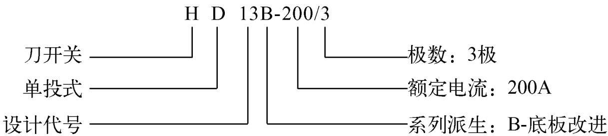


图 1-3　HD 型单投刀开关图形符号（作隔离开关用）

a）一般隔离开关符号　b）手动隔离开关符号　c）三极单投刀隔离开关符号

单投刀开关的型号含义如下：



设计代号：11—中央手柄式，12—侧方正面杠杆操作机构式，13—中央正面杠杆操作机构式，14—侧面手柄式。

（2）HS 型双投刀开关

HS 型双投刀开关也称转换开关，其作用和单投刀开关类似，常用于双电源的切换或双供电线路的切换等，其实物图及图形符号如图 1-4 所示。由于双投刀开关具有机械互锁的结构特点，因此可以防止双电源的并联运行和两条供电线路同时供电。

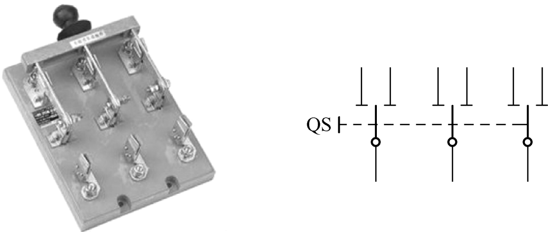


图 1-4　HS 型双投刀开关实物图及图形符号

（3）HR 型熔断器式刀开关

HR 型熔断器式刀开关也称刀熔开关，它实际上是将刀开关和熔断器组合成一体的电器。刀熔开关操作方便，并简化了供电线路，在供配电线路上应用很广泛，其实物图及图形符号如图 1–5 所示。刀熔开关可以切断故障电流，但不能切断正常的工作电流，所以一般应在无正常工作电流的情况下进行操作。

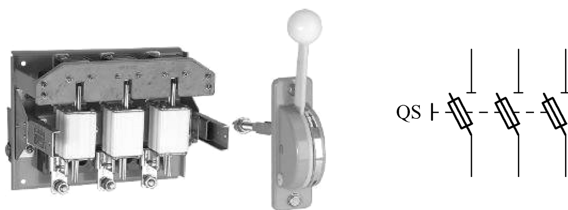


图 1–5　HR 型熔断器式刀开关示意图及图形符号

（4）组合开关

组合开关又称转换开关，控制容量比较小，结构紧凑，常用于空间比较狭小的场所，机床和配电箱等。组合开关一般用于电气设备的非频繁操作、切换电源和负载以及控制小容量感应电动机和小型电器。

组合开关由动触点、静触点、绝缘连杆转轴、手柄、定位机构及外壳等部分组成。其动、静触点分别叠装于数层绝缘壳内，当转动手柄时，每层的动触片随转轴一起转动。

常用的产品有 HZ5、HZ10 和 HZ15 系列。HZ5 系列是类似万能转换开关的产品，其结构与一般转换开关有所不同；组合开关有单极、双极和多极之分。

组合开关的实物图及图形符号如图 1–6 所示。

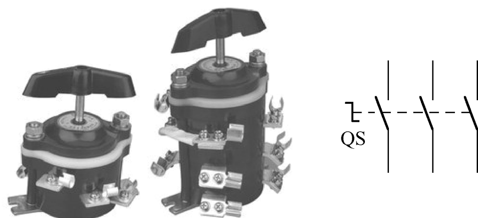


图 1–6　组合开关实物图和图形符号

1.1.2　熔断器

熔断器在电路中主要起短路保护作用，用于保护线路。熔断器的熔体串接于被保护的电路中，熔断器以其自身产生的热量使熔体熔断，从而自动切断电路，实现短路保护及过载保护。熔断器具有结构简单、体积小、重量轻、使用维护方便、价格低廉、分断能力较高、限流能力良好等优点，因此在电路中得到广泛应用。

（1）熔断器的结构原理及分类

熔断器由熔体和安装熔体的绝缘底座（或称熔管）组成。熔体由易熔金属材料铅、锌、锡、铜、银及其合金制成，形状常为丝状或网状。由铅锡合金和锌等低熔点金属制成的熔体，因不易灭弧，多用于小电流电路；由铜、银等高熔点金属制成的熔体，易于灭弧，多用

于大电流电路。

熔断器串接于被保护电路中，电流通过熔体时产生的热量与电流平方和电流通过的时间成正比，电流越大，则熔体熔断时间越短，这种特性称为熔断器的反时限保护特性或安秒特性，如图 1-7 所示。图中 I_N 为熔断器额定电流，熔体允许长期通过额定电流而不熔断。

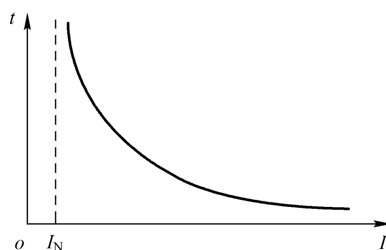


图 1-7　熔断器的反时限保护特性

熔断器种类很多，按结构分为开启式、半封闭式和封闭式；按电流形式分为直流熔断器和交流熔断器按有无填料分为有填料式、无填料式；按用途分为工业用熔断器、保护半导体器件熔断器及自复式熔断器等。

（2）熔断器的主要技术参数

熔断器的主要技术参数包括额定电压、熔体额定电流、熔断器额定电流、极限分断能力等。

① 额定电压：指保证熔断器能长期正常工作的电压。

② 熔体额定电流：指熔体长期通过而不会熔断的电流。

③ 熔断器额定电流：指保证熔断器能长期正常工作的电流。

④ 极限分断能力：指熔断器在额定电压下所能开断的最大短路电流。在电路中出现的最大电流一般是指短路电流值，所以，极限分断能力也反映了熔断器分断短路电流的能力。

（3）常用的熔断器

① 插入式熔断器

插入式熔断器如图 1-8a 所示。常用的产品有 RC1A 系列，主要用于低压分支电路的短路保护，因其分断能力较小，多用于照明电路和小型动力电路中。

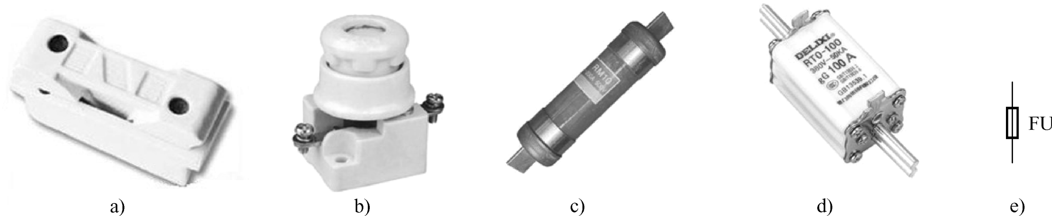


图 1-8　熔断器类型及图形符号

a）插入式　b）螺旋式　c）RM10 型密封管式　d）RT 型有填料密封管式　e）图形符号

② 螺旋式熔断器

螺旋式熔断器如图 1-8b 所示。熔芯内装有熔丝，并填充石英砂，用于熄灭电弧，分断能力强。熔体上的上端盖有一熔断指示器，一旦熔体熔断，指示器马上弹出，可透过瓷帽上的玻璃孔观察到。常用产品有 RL6、RL7 和 RLS2 等系列，其中 RL6 和 RL7 多用于机床配电电路中；RLS2 为快速熔断器，主要用于保护半导体元件。

③ RM10 型密封管式熔断器

RM10 型密封管式熔断器为无填料管式熔断器，如图 1-8c 所示。主要用于供配电系统作为线路的短路保护及过载保护，它采用变截面片状熔体和密封纤维管。由于熔体较窄处的电阻小，在短路电流通过时产生的热量最大，先熔断，因而可产生多个熔断点使电弧分散，

以利于灭弧。短路时其电弧燃烧密封纤维管产生高压气体，以便将电弧迅速熄灭。

④ RT 型有填料密封管式熔断器

RT 型有填料密封管式熔断器如图 1-8d 所示。熔断器中装有石英砂，用于冷却和熄灭电弧，熔体为网状，短路时可使电弧分散，由石英砂将电弧冷却熄灭，可将电弧在短路电流达到最大值之前迅速熄灭，以限制短路电流。此为限流式熔断器，常用于大容量电力网或配电设备中。常用产品有 RTl2、RTl4、RTl5 和 RS3 等系列，RS2 系列为快速熔断器，主要用于保护半导体元件。

（4）熔断器选择

① 低压熔断器的类型选择

选择熔断器可依据负载的保护特性、短路电流的大小和使用场合。一般按电网电压选用相应电压等级的熔断器，按配电系统中可能出现的最大短路电流选择有相应分断能力的熔断器，根据被保护负载的性质和容量选择熔体的额定电流。

② 低压熔断器的容量选择可依据不同的电气设备和线路进行。

a. 照明回路冲击电流很小，所以熔断器的选用系数应尽量小一些。

$$I_{RN} \geqslant I \quad \text{或} \quad I_{RN}=(1.1\sim1.5)I$$

式中　I_{RN}——熔体的额定电流（A）；

I——电器的实际工作电流（A）。

b. 单台电动机负载电气回路中有冲击电流，熔断器的选用系数应尽量大一些。

$$I_{RN} \geqslant (1.5\sim2.5)I$$

c. 多台电动机负载电气回路中，应考虑电动机有同时起动的可能性，所以熔断器的选用应按下列原则选用。

$$I_{RN}=(1.5\sim2.5)I_{Nm}+\Sigma I_N$$

式中　I_{Nm}——设备中最大的一台电动机的额定电流（A）；

I_N——设备中去除最大一台电动机后其他电动机的额定电流之和（A）。

低压熔断器在选用时应严格注意级间的保护原则，切忌发生越级保护的现象，选用中除了依据供电回路短路电阻外，还应适当地考虑上下级的级差，一般级差在 1 ~2 个级差。

1.1.3　断路器

低压断路器俗称自动开关或空气开关，用于低压配电电路中不频繁的通断控制。在电路发生短路、过载或欠电压等故障时能自动分断故障电路，是一种控制兼保护电器。

断路器的种类繁多，按其用途和结构特点可分为 DW 型框架式断路器、DZ 型塑料外壳式断路器、DS 型直流快速断路器和 DWX 型、DWZ 型限流式断路器等。框架式断路器主要用作配电线路的保护开关，而塑料外壳式断路器除可用作配电线路的保护开关外，还可用作电动机、照明电路及电热电路的控制开关。

（1）断路器的结构和工作原理

断路器主要由 3 个基本部分组成，即触点、灭弧系统和各种脱扣器，包括过电流脱扣器、失压（欠电压）脱扣器、热脱扣器、分励脱扣器和自由脱扣器。

图 1-9 是断路器实物图及图形符号。断路器开关是靠操作机构手动或电动合闸的，触点闭合后，自由脱扣机构将触点锁在合闸位置上。当电路发生上述故障时，通过各自的脱扣

器使自由脱扣机构动作，自动跳闸以实现保护作用。分励脱扣器则作为远距离控制分断电路之用。

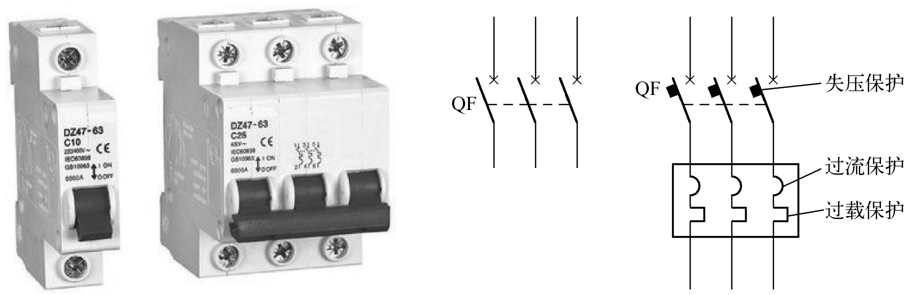


图 1-9　断路器实物图及图形符号

过电流脱扣器用于线路的短路和过电流保护，当线路的电流大于整定的电流值时，过电流脱扣器所产生的电磁力使挂钩脱扣，动触点在弹簧的拉力下迅速断开，实现断路器的跳闸功能。

热脱扣器用于线路的过负载保护，工作原理和热继电器相同。

失压（欠电压）脱扣器用于失压保护，如图 1-9 所示，失压脱扣器的线圈直接接在电源上，处于吸合状态，断路器可以正常合闸；当停电或电压很低时，失压脱扣器的吸力小于弹簧的反力，弹簧使动铁心向上使挂钩脱扣，实现断路器的跳闸功能。

分励脱扣器用于远方跳闸，当在远方按下按钮时，分励脱扣器得电产生电磁力，使其脱扣跳闸。

不同断路器的保护是不同的，使用时应根据需要选用。在图形符号中也可以标注其保护方式，如图 1-9 所示，断路器图形符号中标注了失压、过负载、过电流 3 种保护方式。

（2）低压断路器的选择原则

低压断路器的选择应从以下几方面考虑。

① 断路器类型的选择：应根据使用场合和保护要求来选择。如一般选用塑壳式；短路电流很大时选用限流型；额定电流比较大或有选择性保护要求时选用框架式；控制和保护含有半导体器件的直流电路时应选用直流快速断路器等。

② 断路器额定电压、额定电流应大于或等于线路、设备的正常工作电压、工作电流。

③ 断路器极限通断能力大于或等于电路最大短路电流。

④ 欠电压脱扣器额定电压等于线路额定电压。

⑤ 过电流脱扣器的额定电流大于或等于线路的最大负载电流。

⑥ 低压断路器的容量选择。

低压断路器的容量选择要综合考虑短路、过载时的保护特性。

a. 单台电动机的过流保护应按下式计算：

$$I_{SZD} \geqslant KI_{SN}$$

式中　I_{SZD}——瞬时或短时过电流脱扣器整定电流值（A）；

K——可靠系数，对动作时间大于 0.02 s 的断路器，K 取 1.35，对动作时间小于 0.02 s 的断路器，K 取 1.7～2.0；

I_{SN}——电动机的起动电流（A）。

b. 多台电动机的过流保护应按下式计算：

$$I_{SZD} \geqslant 1.35(I_{SNMAX} + \Sigma I)$$

式中 I_{SNMAX}——最大的电动机起动电流（A）；

ΣI——其余电动机工作电流之和（A）。

c. 单台电动机的过载保护应按下式计算：

$$I_{gzd} > KI_{js}$$

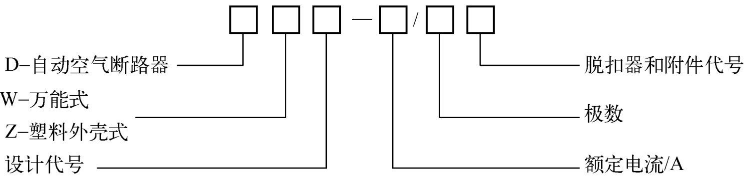
式中 I_{gzd}——过载电流的整定值（A）；

K——可靠系数，一般取 0.9～1.1；

I_{js}——线路的计算电流或实际电流（A）。

（3）低压断路器的型号种类

低压断路器的结构和型号种类很多，目前我国常用的有 DW 和 DZ 系列。DW 型也叫作万能式空气开关，DZ 型叫作塑料外壳式空气开关，其产品代号含义如下：



应注意的是，不同型号的低压断路器分别具有不同的保护机构和参数的整定方法，使用时应根据电路的保护要求选择其型号并进行参数的整定。

1.1.4 接触器

接触器主要用于控制电动机、电热设备、电焊机、电容器组等，能频繁地接通或断开交直流主电路，实现远距离自动控制。它具有低电压释放保护功能，在电力拖动自动控制线路中被广泛应用。

接触器有交流接触器和直流接触器两大类型。下面介绍交流接触器。

图 1–10 所示为交流接触器的结构示意图及图形符号。

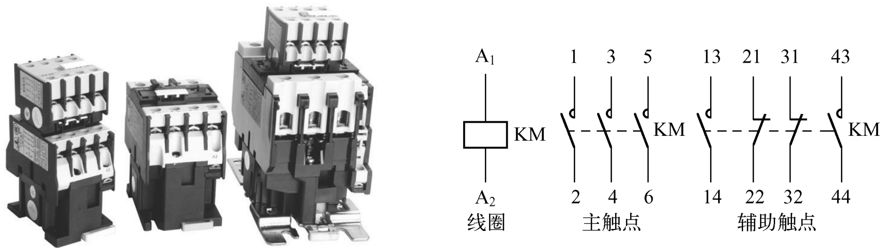


图 1–10 交流接触器及图形符号

（1）交流接触器的组成部分

① 电磁机构：电磁机构由线圈、动铁心（衔铁）和静铁心组成。

② 触点系统：交流接触器的触点系统包括主触点和辅助触点。主触点用于通断主电路，有 3 对或 4 对常开触点；辅助触点用于控制电路，起电气联锁或控制作用，通常有两对常开触点两对常闭触点。

③ 灭弧装置：容量在 10A 以上的接触器都有灭弧装置。对于小容量的接触器，常采用双断口桥形触点以利于灭弧；对于大容量的接触器，常采用纵缝灭弧罩及栅片灭弧结构。

④ 其他部件：包括反作用弹簧、缓冲弹簧、触点压力弹簧、传动机构及外壳等。

接触器上标有端子标号，线圈为 A_1、A_2，主触点 1、3、5 接电源侧，2、4、6 接负载侧。辅助触点用两位数表示，前一位为辅助触点顺序号，后一位的 3、4 表示常开触点，1、2 表示常闭触点。

接触器的控制原理很简单，当线圈接通额定电压时，产生电磁力，克服弹簧反力，吸引动铁心向下运动，动铁心带动绝缘连杆和动触点向下运动使常开触点闭合，常闭触点断开。当线圈失电或电压低于释放电压时，电磁力小于弹簧反力，常开触点断开，常闭触点闭合。

（2）接触器的主要技术参数和类型

① 额定电压：接触器的额定电压是指主触点的额定电压。交流有 220 V、380 V 和 660 V，在特殊场合应用的额定电压高达 1140 V，直流主要有 110 V、220 V 和 440 V。

② 额定电流：接触器的额定电流是指主触点的额定工作电流。它是在一定的条件（额定电压、使用类别和操作频率等）下规定的，目前常用的电流等级为 10 A ~ 800 A。

③ 吸引线圈的额定电压：交流有 36 V、127 V、220 V 和 380 V，直流有 24 V、48 V、220 V和 440 V。

④ 机械寿命和电气寿命：接触器是频繁操作电器，应有较高的机械和电气寿命，该指标是产品质量的重要指标之一。

⑤ 额定操作频率：接触器的额定操作频率是指每小时允许的操作次数，一般为 300 次/小时、600 次/小时和 1200 次/小时。

⑥ 动作值：动作值是指接触器的吸合电压和释放电压。规定接触器的吸合电压大于线圈额定电压的 85% 时应可靠吸合，释放电压不高于线圈额定电压的 70% 。

常用的交流接触器有 CJ10、CJ12、CJ10X、CJ20、CJX1、CJX2、3TB 和 3TD 等系列。

（3）接触器的选择

① 根据负载性质选择接触器的类型。

② 额定电压应大于或等于主电路工作电压。

③ 额定电流应大于或等于被控电路的额定电流。对于电动机负载，还应根据其运行方式适当增大或减小。

④ 吸引线圈的额定电压与频率要与所在控制电路的选用电压和频率相一致。

1. 1. 5　控制继电器

控制继电器用于电路的逻辑控制，继电器具有逻辑记忆功能，能组成复杂的逻辑控制电路，继电器用于将某种电量（如电压、电流）或非电量（如温度、压力、转速、时间等）的变化量转换为开关量，以实现对电路的自动控制功能。

继电器的种类很多，按输入量可分为电压继电器、电流继电器、时间继电器、速度继电器、压力继电器等；按工作原理可分为电磁式继电器、感应式继电器、电动式继电器、电子式继电器等；按用途可分为控制继电器、保护继电器等；按输入量变化形式可分为有无继电器和量度继电器。

（1）电磁式继电器

在控制电路中用的继电器大多数是电磁式继电器。电磁式继电器具有结构简单、价格低廉、使用维护方便、触点容量小（一般在 5 A 以下）、触点数量多且无主辅之分、无灭弧装置、体积小、动作迅速、准确、控制灵敏、可靠等特点，广泛地应用于低压控制系统中。常用的电磁式继电器有电流继电器、电压继电器、中间继电器以及各种小型通用继电器等。

电磁式继电器的结构和工作原理与接触器相似，主要由电磁机构和触点组成。电磁式继电器也有直流和交流两种。

继电器的主要特性是输入 – 输出特性，又称为继电特性，如图 1–11 所示。

图 1–11　继电器输入 – 输出特性

当继电器输入量 X 由 0 增加至 X_2 之前，输出量 Y 为 0。当输入量增加到 X_2 时，继电器吸合，输出量 Y 为 1，表示继电器线圈得电，常开触点闭合，常闭触点断开。当输入量继续增大时，继电器动作状态不变。

当输出量 Y 为 1 的状态下，输入量 X 减小，当小于 X_2 时 Y 值仍不变，当 X 再继续减小至小于 X_1 时，继电器释放，输出量 Y 变为 0，X 再减小，Y 值仍为 0。

在继电特性曲线中，X_2 称为继电器吸合值，X_1 称为继电器释放值。$k = X_1/X_2$，称为继电器的返回系数，它是继电器的重要参数之一。

返回系数 k 值可以调节，不同场合对 k 值的要求不同。例如一般控制继电器要求 k 值低些，在 0.1 ~ 0.4 之间，这样继电器吸合后，输入量波动较大时不致引起误动作。保护继电器要求 k 值高些，一般在 0.85 ~ 0.9 之间。k 值是反映吸力特性与反力特性配合紧密程度的一个参数，一般 k 值越大，继电器灵敏度越高，k 值越小，灵敏度越低。

（2）中间继电器

中间继电器是最常用的继电器之一，它的结构和接触器基本相同，如图 1–12 左图所示，其图形符号如图 1–12 右图所示。

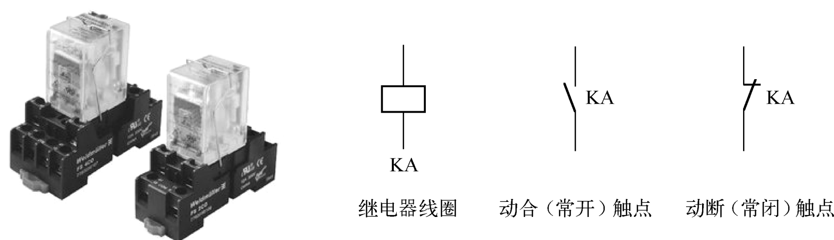


图 1–12　中间继电器及图形符号

中间继电器在控制电路中起逻辑变换和状态记忆的功能，以及用于扩展触点的容量和数量。另外，在控制电路中还可以调节各继电器、开关之间的动作时间，防止电路误动作的作用。中间继电器实质上是一种电压继电器，它是根据输入电压的有或无而动作的，一般触点对数多，触点容量额定电流为 5 A ~ 10 A。中间继电器体积小，动作灵敏度高，一般不用于直接控制电路的负载，但当电路的负载电流在 5 A ~ 10 A 时，也可代替接触器起控制负载的

作用。中间继电器的工作原理和接触器一样，触点较多，一般为四常开触点和四常闭触点。

常用的中间继电器型号有 JZ7、JZ14 等。

（3）电流继电器

电流继电器的输入量是电流，它是根据输入电流大小而动作的继电器。电流继电器的线圈串入电路中，以反映电路电流的变化，其线圈匝数少、导线粗、阻抗小。电流继电器可分为欠电流继电器和过电流继电器。

欠电流继电器用于欠电流保护或控制，如直流电动机励磁绕组的弱磁保护、电磁吸盘中的欠电流保护、绕线式异步电动机起动时电阻的切换控制等。欠电流继电器在电路正常工作时处于吸合动作状态，常开触点处于闭合状态，常闭触点处于断开状态，当电路出现不正常现象或故障现象导致电流下降或消失时，继电器中流过的电流小于释放电流而动作；过电流继电器用于过电流保护或控制，如起重机电路中的过电流保护。过电流继电器在电路正常工作时流过正常工作电流，正常工作电流小于继电器所整定的动作电流，继电器不动作，当电流超过动作电流整定值时才动作。过电流继电器动作时其常开触点闭合，常闭触点断开。

电流继电器作为保护电器时，其图形符号如图 1-13 所示。

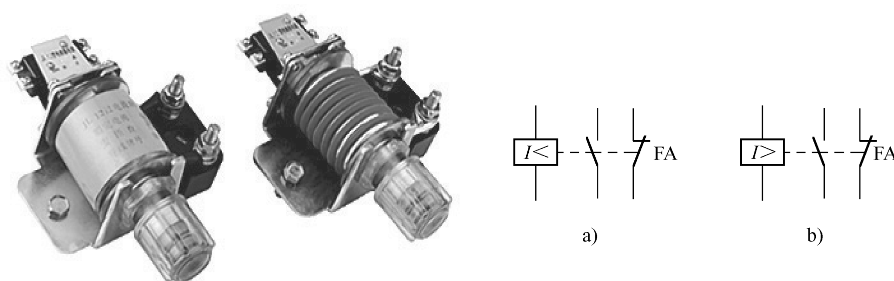


图 1-13　电流继电器实物图及图形符号

a）欠电流继电器　b）过电流继电器

（4）电压继电器

电压继电器的输入量是电路的电压大小，根据输入电压大小而动作。电压继电器工作时并联在电路中，因此线圈匝数多、导线细、阻抗大，反映电路中电压的变化，用于电路的电压保护。与电流继电器类似，电压继电器也分为欠电压继电器和过电压继电器两种。

过电压继电器动作电压范围为（105% ~120%）U_N；欠电压继电器吸合电压动作范围为（20% ~50%）U_N，释放电压调整范围为（7% ~20%）U_N；零电压继电器当电压降低至（5% ~25%）U_N时动作，它们分别起过压、欠压、零压保护。电压继电器常用在电力系统继电保护中，在低压控制电路中使用较少。

电压继电器作为保护电器时，其图形符号如图 1-14 所示。

（5）时间继电器

时间继电器在控制电路中用于时间的控制。其种类很多，按其动作原理可分为电磁式、空气阻尼式、电动式和电子式等；按延时方式可分为通电延时型和断电延时型。下面以 JS7 型空气阻尼式时间继电器为例说明其工作原理。

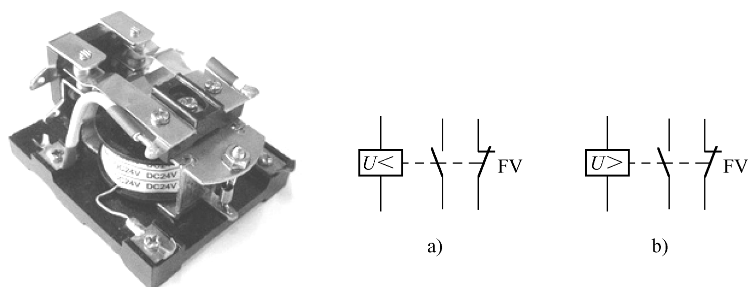


图 1-14　电压继电器的图形符号
a）欠电压继电器　b）过电压继电器

空气阻尼式时间继电器是利用空气阻尼原理获得延时的，它由电磁机构、延时机构和触点系统三部分组成。电磁机构为直动式双 E 型铁心，触点系统借用 LX5 型微动开关，延时机构采用气囊式阻尼器。

空气阻尼式时间继电器可以做成通电延时型，也可改成断电延时型，电磁机构可以是直流的，也可以是交流的，如图 1-15 所示。

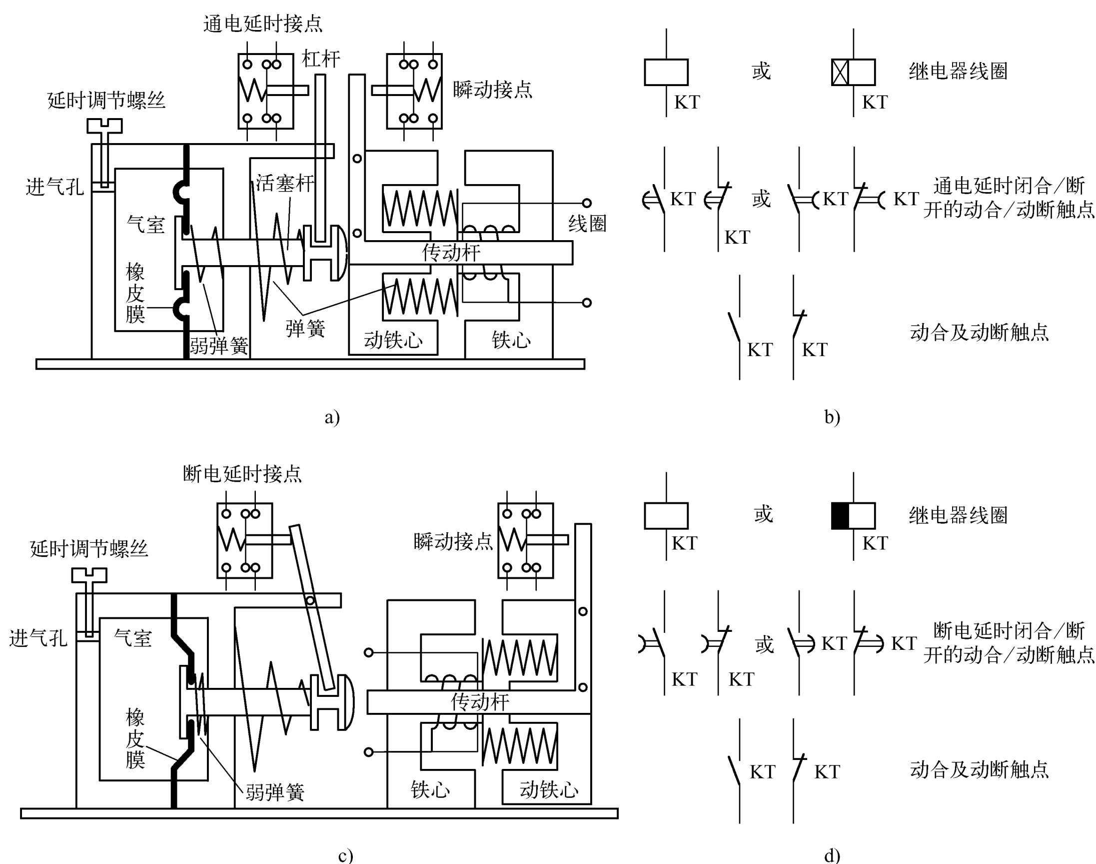


图 1-15　空气阻尼式时间继电器示意图及图形符号
a）通电延时继电器示意图　b）通电延时继电器图形符号
c）断电延时继电器示意图　d）断电延时继电器图形符号

现以通电延时型时间继电器为例介绍其工作原理。

图1-15a中通电延时型时间继电器为线圈不得电时的情况，当线圈通电后，动铁心吸合，带动L型传动杆向右运动，使瞬动接点受压，其接点瞬时动作。活塞杆在塔形弹簧的作用下，带动橡皮膜向右移动，弱弹簧将橡皮膜压在活塞上，橡皮膜左方的空气不能进入气室，形成负压，只能通过进气孔进气，因此活塞杆只能缓慢地向右移动，其移动的速度和进气孔的大小有关（通过延时调节螺丝调节进气孔的大小可改变延时时间）。经过一定的延时后，活塞杆移动到右端，通过杠杆压动微动开关（通电延时接点），使其常闭触点断开，常开触点闭合，起到通电延时作用。

当线圈断电时，电磁吸力消失，动铁心在反力弹簧的作用下释放，并通过活塞杆将活塞推向左端，这时气室内中的空气通过橡皮膜和活塞杆之间的缝隙排掉，瞬动接点和延时接点迅速复位，无延时。

如果将通电延时型时间继电器的电磁机构反向安装，就可以改为断电延时型时间继电器，如图1-15c中断电延时型时间继电器所示。线圈不得电时，塔形弹簧将橡皮膜和活塞杆推向右侧，杠杆将延时接点压下（注意，原来通电延时的常开触点现在变成了断电延时的常闭触点了，原来通电延时的常闭触点现在变成了断电延时的常开触点），当线圈通电时，动铁心带动L型传动杆向左运动，使瞬动接点瞬时动作，同时推动活塞杆向左运动，如前所述，活塞杆向左运动不延时，延时接点瞬时动作。线圈失电时动铁心在反力弹簧的作用下返回，瞬动接点瞬时动作，延时接点延时动作。

时间继电器线圈和延时接点的图形符号都有两种画法，线圈中的延时符号可以不画，接点中的延时符号可以画在左边也可以画在右边，但是圆弧的方向不能改变，如图1-15b和d所示。

空气阻尼式时间继电器的优点是结构简单、延时范围大、寿命长、价格低廉，且不受电源电压及频率波动的影响，其缺点是延时误差大、无调节刻度指示，一般适用延时精度要求不高的场合。常用的产品有JS7－A、JS23等系列，其中JS7－A系列的主要技术参数为延时范围，分0.4 s～60 s和0.4s～180 s两种，操作频率为600次/小时，触点容量为5 A，延时误差为±15%。在使用空气阻尼式时间继电器时，应保持延时机构的清洁，防止因进气孔堵塞而失去延时作用。

时间继电器在选用时应根据控制要求选择其延时方式，根据延时范围和精度选择继电器的类型。

（6）热继电器

热继电器主要是用于电气设备（主要是电动机）的过负载保护。热继电器是一种利用电流热效应原理工作的电器，它具有与电动机容许过载特性相近的反时限动作特性，主要与接触器配合使用，用于对三相异步电动机的过负载和断相保护。

三相异步电动机在实际运行中，常会遇到因电气或机械原因等引起的过电流（过载和断相）现象。如果过电流不严重，持续时间短，绕组不超过允许温升，这种过电流是允许的；如果过电流情况严重，持续时间较长，则会加快电动机绝缘老化，甚至烧毁电动机，因此，在电动机回路中应设置电动机保护装置。常用的电动机保护装置种类很多，使用最多、最普遍的是双金属片式热继电器。目前，双金属片式热继电器均为三相式，有带断相保护和不带断相保护两种。

① 热继电器的工作原理

图 1-16a 所示是双金属片式热继电器的结构示意图，图 1-16b 所示是其图形符号。由图可见，热继电器主要由双金属片、热元件、复位按钮、传动杆、拉簧、调节旋钮、复位螺丝、触点和接线端子等组成。

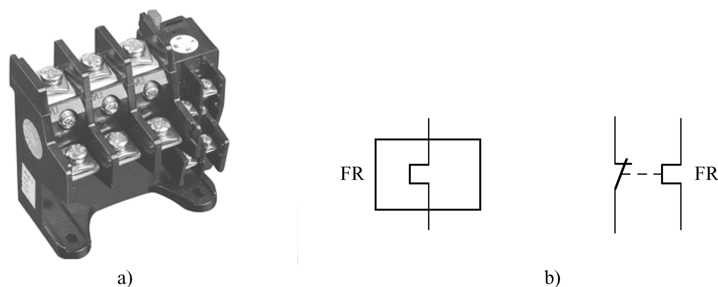


图 1-16　热继电器及图形符号
a）结构示意图　b）图形符号

双金属片是一种将两种线膨胀系数不同的金属用机械辗压方法使之形成一体的金属片。膨胀系数大的（如铁镍铬合金、铜合金或高铝合金等）称为主动层，膨胀系数小的（如铁镍类合金）称为被动层。由于两种线膨胀系数不同的金属紧密地贴合在一起，当产生热效应时，使得双金属片向膨胀系数小的一侧弯曲，由弯曲产生的位移带动触点动作。

热元件一般由铜镍合金、镍铬铁合金或铁铬铝等合金电阻材料制成，其形状有圆丝、扁丝、片状和带材几种。热元件串接于电机的定子电路中，通过热元件的电流就是电动机的工作电流（大容量的热继电器装有速饱和互感器，热元件串接在其二次回路中）。当电动机正常运行时，其工作电流通过热元件产生的热量不足以使双金属片变形，热继电器不会动作。当电动机发生过电流且超过整定值时，双金属片的热量增大而发生弯曲，经过一定时间后，使触点动作，通过控制电路切断电动机的工作电源。同时，热元件也因失电而逐渐降温，经过一段时间的冷却，双金属片恢复到原来状态。

热继电器动作电流的调节是通过旋转调节旋钮来实现的。调节旋钮为一个偏心轮，旋转调节旋钮可以改变传动杆和动触点之间的传动距离，距离越长动作电流就越大，反之动作电流就越小。

热继电器复位方式有自动复位和手动复位两种，将复位螺丝旋入，使常开的静触点向动触点靠近，这样动触点在闭合时处于不稳定状态，在双金属片冷却后动触点也返回原位，为自动复位方式。如将复位螺丝旋出，触点不能自动复位，为手动复位置方式。在手动复位置方式下，需在双金属片恢复原状时按下复位按钮才能使触点复位。

② 热继电器的选择原则

热继电器主要用于电动机的过载保护，使用中应考虑电动机的工作环境、起动情况、负载性质等因素，具体应按以下几个方面来选择。

a. 热继电器结构形式的选择：星形接法的电动机可选用两相或三相结构热继电器，三角形接法的电动机应选用带断相保护装置的三相结构热继电器。

b. 热继电器的动作电流整定值一般为电动机额定电流的 1.05 ~1.1 倍。

c. 对于重复短时工作的电动机（如起重机电动机），由于电动机不断重复升温，热继电器双金属片的温升跟不上电动机绕组的温升，电动机将得不到可靠的过载保护。因此，不宜选用双金属片热继电器，而应选用过电流继电器或能反映绕组实际温度的温度继电器来进行保护。

(7) 速度继电器

速度继电器又称为反接制动继电器，主要用于三相笼型异步电动机的反接制动控制。图 1-17 为速度继电器的原理示意图及图形符号，它主要由转子、定子和触点 3 部分组成。

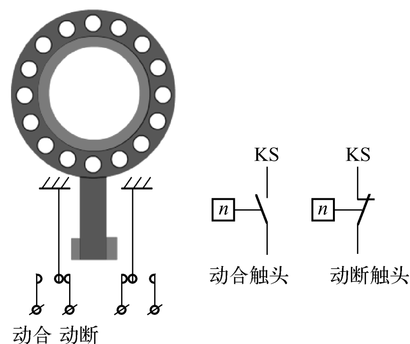


图 1-17　速度继电器的原理示意图及图形符号

转子是一个圆柱形永久磁铁，定子是一个笼型空心圆环，由硅钢片叠成，并装有笼型绕组。其转子的轴与被控电动机的轴相连接，当电动机转动时，转子（圆柱形永久磁铁）随之转动产生一个旋转磁场，定子中的笼型绕组切割磁力线而产生感应电流和磁场，两个磁场相互作用，使定子受力而跟随转动，当达到一定转速时，装在定子轴上的摆锤推动簧片触点运动，使常闭触点断开，常开触点闭合。当电动机转速低于某一数值时，定子产生的转矩减小，触点在簧片作用下复位。

常用的速度继电器有 JYl 型和 JFZ0 型两种。其中 JYl 型可在 700～3600 r/min 范围工作，JFZ0－1 型适用于 300～1000 r/min，JFZ0－2 型适用于 1000～3000 r/min。

一般速度继电器都具有两对转换触点，一对用于正转时动作，另一对用于反转时动作。触点额定电压为 380 V，额定电流为 2 A。通常速度继电器动作转速为 130 r/min，复位转速在 100 r/min 以下。

(8) 液位继电器

液位继电器主要用于对液位的高低进行检测并发出开关量信号，以控制电磁阀、液泵等设备对液位的高低进行控制。液位继电器的种类很多，工作原理也不尽相同，下面介绍 JYF－02 型液位继电器。其结构示意图及图形符号如图 1-18 所示。浮筒置于液体内，浮筒的另一端为一根磁钢，靠近磁钢的液体外壁也装一根磁钢，并和动触点相连，当水位上升时，受浮力上浮而绕固定支点上浮，带动磁钢条向下，当内磁钢 N 极低于外磁钢 N 极时，由于液体壁内外两根磁钢同性相斥，壁外的磁钢受排斥力迅速上翘，带动触点迅速动作。同理，当液位下降，内磁钢 N 极高于外磁钢 N 极时，外磁钢受排斥力迅速下翘，带动触点迅速动作。液位高低的控制是由液位继电器安装的位置来决定的。

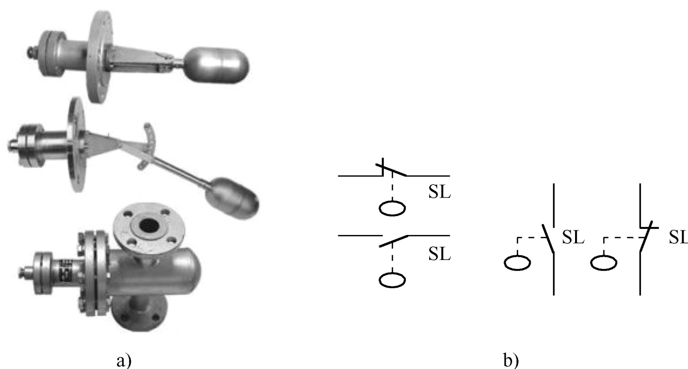


图 1-18　JYF－02 型液位继电器实物图及图形符号

a) 液位继电器实物　b) 图形符号

（9）压力继电器

压力继电器主要用于对液体或气体压力的高低进行检测并发出开关量信号，以控制电磁阀、液泵等设备对压力的高低进行控制。图 1–19 为压力继电器结构示意图及图形符号。



图 1–19　压力继电器实物图及图形符号

a）压力继电器实物　b）图形符号

压力继电器主要由压力传送装置和微动开关等组成，液体或气体压力经压力入口推动橡皮膜和滑杆，克服弹簧反力向上运动，当压力达到给定压力时，触动微动开关，发出控制信号，旋转调压螺母可以改变给定压力。

1. 1. 6　主令电器

主令电器在控制电路中主要是用于发布控制命令，其作用是实现远程操作和自动控制。常用的主令电器有控制按钮、行程开关、接近开关，万能转换开关，主令控制器有脚踏开关、倒顺开关、紧急开关、钮子开关等。

（1）控制按钮

控制按钮一般和接触器或继电器配合使用，实现对电动机的远程操作、控制电路的电气联锁等。它是一种结构简单、使用广泛的手动主令电器。控制按钮的结构由按钮帽、复位弹簧、桥式触点和外壳等组成，如图 1–20 所示。

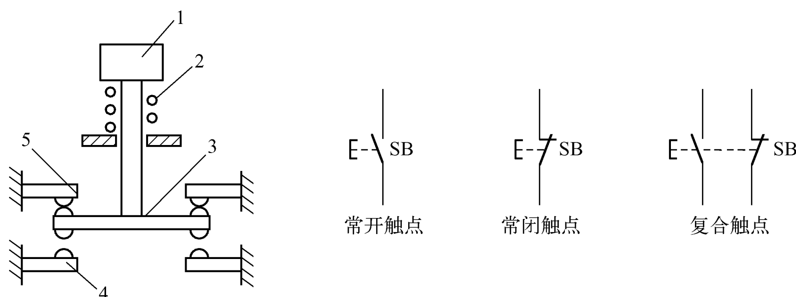


图 1–20　控制按钮原理图及图例符号

1—按钮帽　2—复位弹簧　3—动触点　4—常开静触点　5—常闭静触点

控制按钮通常配备一个常开触点和一个常闭触点（也可以进行多组触点的扩展），当控制按钮被按下时，桥式动触点将常闭静触点断开，常开静触点闭合。释放后，弹簧将桥式动触点拉回原位，相应的触点也复位。

① 常开按钮是用于控制电动机和控制电路的起动和运行开始。使用时一般只对其常开触点进行接线，常开按钮通常选其颜色为绿色，安装时布局在上方或是左侧。

② 常闭按钮是用于控制电动机和控制电路的停止。使用时一般只对其常闭触点进行接线，常闭按钮通常选其颜色为红色，安装时布局在下方或是右侧。

（2）行程开关

行程开关又叫作限位开关，它的种类很多，按运动形式可分为直动式、微动式、转动式等；按触点的性质可分为有触点式和无触点式。

① 有触点行程开关

有触点行程开关简称行程开关，行程开关的工作原理和按钮相同，区别在于它不是靠手的按压，而是利用生产机械运动的部件碰压而使触点动作来发出控制指令的主令电器。它用于控制生产机械的运动方向、速度、行程大小或位置等，其结构形式多种多样。

图 1-21 所示为几种操作类型的行程开关及图形符号。

行程开关的主要参数有型式、动作行程、工作电压及触点的电流容量。目前国内生产的行程开关有 LXK3、3SE3、LXl9、LXW 和 LX 等系列。

常用的行程开关有 LX19、LXW5、LXK3、LX32 和 LX33 等系列。

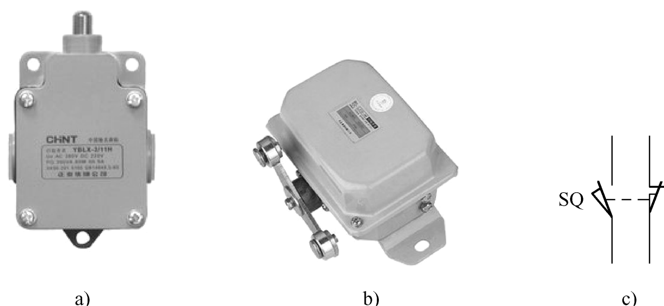


图 1-21　行程开关实物图及图形符号

a）直动式行程开关　b）微动式行程开关　c）图形符号

② 无触点行程开关

无触点行程开关又称接近开关，它可以代替有触点行程开关来完成行程控制和限位保护，还可用于高频计数、测速、液位控制、零件尺寸检测、加工程序的自动衔接等的非接触式开关。由于它具有非接触式触发、动作速度快、可在不同的检测距离内动作、发出的信号稳定无脉动、工作稳定可靠、寿命长、重复定位精度高以及能适应恶劣的工作环境等特点，所以在机床、纺织、印刷、塑料等工业生产中应用广泛。

无触点行程开关分为有源型和无源型两种，多数无触点行程开关为有源型，主要包括检测元件、放大电路、输出驱动电路 3 部分，一般采用 5 V ~ 24 V 的直流电流，或 220 V 交流电源等。图 1-22 所示为三线式有源型接近开关结构框图。

接近开关按检测元件工作原理可分为高频振荡型、超声波型、电容型、电磁感应型、永磁型、霍尔元件型与磁敏元件型等。不同型式的接近开关所检测的被检测体不同。



图 1-22　有源型接近开关结构框图

电容式接近开关可以检测各种固体、液体或粉状物体，其主要由电容式振荡器及电子电路组成，它的电容位于传感界面，当物体接近时，将因改变了电容值而振荡，从而产生输出信号。

霍尔接近开关用于检测磁场，一般用磁钢作为被检测体。其内部的磁敏感器件仅对垂直于传感器端面的磁场敏感，当磁极 S 极正对接近开关时，接近开关的输出产生正跳变，输出为高电平，若磁极 N 极正对接近开关时，输出为低电平。

超声波接近开关适于检测不能或不可触及的目标，其控制功能不受声、电、光等因素干扰，检测物体可以是固体、液体或粉末状态的物体，只要能反射超声波即可。其主要由压电陶瓷传感器、发射超声波和接收反射波用的电子装置及调节检测范围用的程控桥式开关等几个部分组成。

高频振荡式接近开关用于检测各种金属，主要由高频振荡器、集成电路或晶体管放大器和输出器 3 部分组成，其基本工作原理是当有金属物体接近振荡器的线圈时，该金属物体内部产生的涡流将吸取振荡器的能量，致使振荡器停振。振荡器的振荡和停振这两个信号，经整形放大后转换成开关信号输出。

接近开关输出形式有两线、三线和四线式几种，晶体管输出类型有 NPN 和 PNP 两种，外形有方型、圆型、槽型和分离型等多种，图 1-23 为槽型三线式 NPN 型光电式接近开关和远距分离型光电开关。

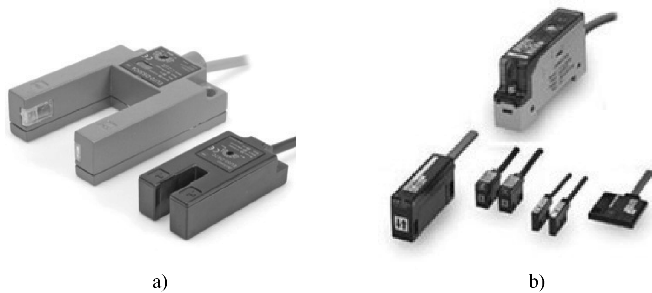


图 1-23　槽型和分离型光电开关

a）槽型光电式接近开关　b）远距分离型光电开关

接近开关的主要参数有型式、动作距离范围、动作频率、响应时间、重复精度、输出型式、工作电压及输出触点的容量等。接近开关的图形符号可用图 1-24 表示。

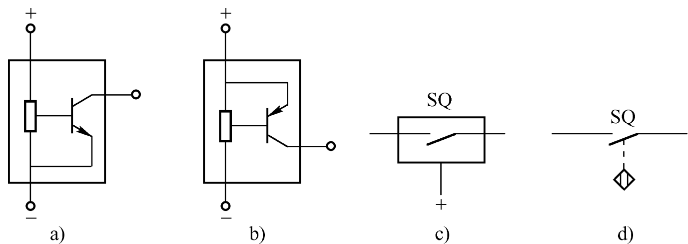


图 1-24　接近开关的图形符号

a）NPM 型　b）PNP 型　c）有源接近开关　d）无源接近开关

接近开关的产品种类十分丰富，常用的国产接近开关有 LJ、3SG 和 LXJ18 等多种系列，国外进口及引进产品亦在国内有大量的应用。

（3）万能转换开关

万能转换开关是一种多档式、控制多回路的主令电器。它主要用于完成对电路的选择控

制、信号转换、电源的换相测量等任务，如手动、自动的切换，多路信号的输入选择，电流表和电压表的换相测量等。结构原理如图 1–25 所示。



LW5–15D0403/2				
触头编号		45°	0°	45°
NO	1–2	X		
NO	3–4	X		
NO	5–6	X	X	
NO	7–8			X

图 1–25　万能转换开关结构图

图 1–25 中万能转换开关打向左 45°时，触点 1 –2、3 –4、5 –6 闭合，触点 7 –8 打开；打向 0°时，只有触点 5 –6 闭合，右 45°时，触点 7 –8 闭合，其余打开。

（4）信号灯

信号灯是用于指示电气运行状态、生产节拍、机械位置、控制命令等的电器器件。其发光源有白炽灯、氖炮、LED 发光元件等形式，通常在低电压中用白炽灯和 LED 发光元件，而在高压中用氖炮。可以单独使用，也可以和按钮组合使用。

信号灯的图形符号如图 1–26 所示。

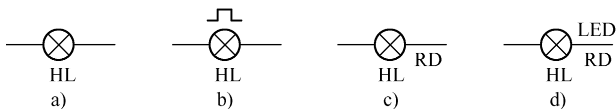


图 1–26　信号灯的图形符号

a）平光灯　b）闪光灯　c）红灯　d）红色发光二极管灯

如果要在图形符号上标注信号灯的颜色，可在靠近图形处标出对应颜色的字母。

红色：RD；黄色：YE；绿色：GN；蓝色：BU；白色：WH。

常用的信号灯型号有 AD11、AD30、ADJ1 等，信号灯的主要参数有工作电压、安装尺寸及发光颜色等。指示灯的颜色及其含义如表 1–1 所示。

表 1–1　指示灯的颜色及其含义

颜　色	含　　义	说　　明	典 型 应 用
红色	危险告急	可能出现危险和需要立即处理	温度超过规定（或安全）限制 设备的重要部分已被保护电器切断 润滑系统失压有触及带电或运动部件的危险
黄色	注意	情况有变化或即将发生变化	温度（或压力）异常 当仅能承受允许的短时过载
绿色	安全	正常或允许进行	冷却通风正常 自动控制系统运行正常 机器准备起动
蓝色	按需要指定用意	除红、黄、绿三色外的任何指定用意	遥控指示 选择开关在设定位置
白色	无特定用意	任何用意。不能确切地用红黄绿时，以及用作执行时	

(5) 报警器

常用的报警器有电铃和电喇叭等，一般电铃用于正常的操作信号（如设备起动前的警示）和设备的异常现象（如变压器的过载、漏油）。电喇叭用于设备的故障信号（如线路短路跳闸）。报警器的图形符号如图1-27所示。

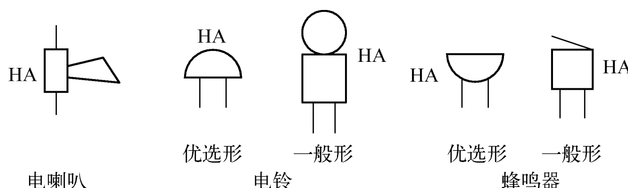


图1-27　报警器的图形符号（符合GB4728.8-84）

1.2　电气图中的图形符号和文字符号

1.2.1　电器的文字符号

电器的文字符号目前执行国家标准GB 5094—85《电气技术中的项目代号》和GB 7159—87《电气技术中的文字符号制定通则》。这两个标准都是根据IEC国际标准而制定的。

在GB 7159—87《电气技术中的文字符号制定通则》中将所有的电气设备、装置和元件分成23个大类，每个大类用一个大写字母表示。文字符号分为基本文字符号和辅助文字符号。

基本文字符号分为单字母符号和双字母符号两种。单字母符号应优先采用，每个单字母符号表示一个电器大类，如表1-2所示。如C表示电容器类，R表示电阻器类等。

双字母符号由一个表示种类的单字母符号和另一个字母组成，第一个字母表示电器的大类，第二个字母表示对某电器大类的进一步划分。例如G表示电源大类，GB表示蓄电池，S表示控制电路开关，SB表示按钮，SP表示压力传感器（继电器）。

文字符号用于标明电器的名称、功能、状态和特征。同一电器如果功能不同，其文字符号也不同，例如照明灯的文字符号为EL，信号灯的文字符号为HL。

辅助文字符号表示电气设备、装置和元件的功能、状态和特征，由1～3位英文名称缩写的大写字母表示，例如辅助文字符号BW（Backward的缩写）表示向后，P（Pressure的缩写）表示压力。辅助文字符号可以和单字母符号组合成双字母符号，例如单字母符号K（表示继电器接触器大类）和辅助文字符号AC（交流）组合成双字母符号KA，表示交流继电器；单字母符号M（表示电动机大类）和辅助文字符号SYN（同步）组合成双字母符号MS，表示同步电动机。辅助文字符号可以单独使用，例如图1-26中的RD表示信号灯为红色。

1.2.2　电器的图形符号

电器的图形符号目前执行国家标准GB 4728—85《电气图用图形符号》，也是根据IEC

国际标准制定的。该标准给出了大量的常用电器图形符号，表示产品特征。通常用比较简单的电器作为一般符号。对于一些组合电器，不必考虑其内部细节时可用方框符号表示，见表 1-2 中的整流器、逆变器、滤波器等。

国家标准 GB 4728—85 的一个显著特点就是图形符号可以根据需要进行组合，在该标准中除了提供了大量的一般符号之外，还提供了大量的限定符号和符号要素，限定符号和符号要素不能单独使用，它相当于一般符号的配件。将某些限定符号或符号要素与一般符号进行组合就可组成各种电气图形符号，例如图 1-9 所示的断路器的图形符号就是由多种限定符号、符号要素和一般符号组合而成的，如图 1-28 所示。

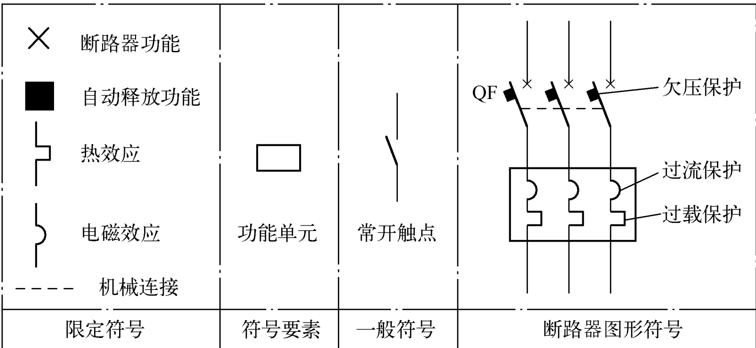


图 1-28 断路器图形符号的组成

表 1-2 常用电器分类及图形符号、文字符号举例

分 类	名 称	图形符号文字符号
A 组件部件	起动装置	A SB_1 SB_2 KM KM HL
B 将电量变换成非电量，将非电量变换成电量	扬声器	B（将电量变换成非电量）
	传声器	B（将非电量变换成电量）
C 电容器	一般电容器	C
	极性电容器	+ C
	可变电容器	C
D 二进制元件	与门	D &
	或门	D ≥1
	非门	D
E 其他	照明灯	EL
F 保护器件	欠电流继电器	I< FA
	过电流继电器	I> FA
	过电压继电器	U< FV

（续）

分　类	名　称	图形符号文字符号
F 保护器件	热继电器	FR FR FR FR FR
	熔断器	FU
G 发生器，发电机，电源	交流发电机	G ~
	直流发电机	G
	电池	GB − +
H 信号器件	电喇叭	HA
	蜂鸣器	HA
	信号灯	HL
I		（不使用）
J		（不使用）
K 继电器，接触器	中间继电器	KA KA
	接触器	KM KM
	通电延时型时间继电器	或 KT KT KT KT 或 KT KT
	断电延时型时间继电器	或 KT KT KT KT 或 KT KT

分　类	名　称	图形符号文字符号
L 电感器，电抗器	电感器	L （一般符号） L （带磁芯符号）
	可变电感器	L
	电抗器	L
M 电动机	笼型电动机	U V W M 3~
	绕线型电动机	U V W M 3~
	他励直流电动机	M
	并励直流电动机	M
	串励直流电动机	M
	三相步进电动机	M
	永磁直流电动机	M
N 模拟元件	运算放大器	N ▷ ∞ − + +

（续）

分　类	名　称	图形符号文字符号
N 模拟元件	反相放大器	N ▷ 1 + −
	数－模转换器	N #/U
N	模－数转换器	N U/#
O		（不使用）
P 测量设备，试验设备	电流表	PA A
	电压表	PV V
	有功能率表	kW PW
	有功电度表	kWh PJ
Q 电力电路的开关器件	断路器	QF
	隔离开关	QS
	刀熔开关	QS
	手动开关	QS QS
	双投刀开关	QS
	组合开关旋转开关	QS
Q 电力电路的开关器件	负载开关	QL
R 电阻器	电阻	R
	固定抽头电阻	R
	可变电阻	R
	电位器	RP
	频敏变阻器	RF
S 控制、记忆、信号电路开关器件选择器	按钮	SB
	急停按钮	SB
	行程开关	SQ
	压力继电器	P SP P
	液位继电器	SL SL SL SL
	速度继电器	SV n SV n SV
	选择开关	SA
	接近开关	SQ
	万能转换开关，凸轮控制器	SA 2 1 0 1 2 1 2 3 4

（续）

分　类	名　称	图形符号文字符号	分　类	名　称	图形符号文字符号
T 变压器互感器	单相变压器	T	X 端子插头插座	插头	优选型　其他型　XP
	自耦变压器	形成1　形成2　T		插座	优选型　其他型　XS
	三相变压器（星形/三角形接线）	形成1　形成2　T		插头插座	优选型　其他型　X
	电压互感器	电压互感器与变压器图形符号相同，文字符号为 TV		连接片	断开时　接通时　XB
	电流互感器	形成1　形成2　TA	Y 电器操作的机械器件	电磁铁	或　YA
U 调制器变换器	整流器	U			
	桥式全波整流器	U		电磁吸盘	或　YH
	逆变器	U		电磁制动器	M　YB
	变频器	f1 f2　U			
V 电子管晶体管	二极管	V		电磁阀	或　或　YV
	晶体管	V	Z 滤波器、限幅器、均衡器、终端设备	滤波器	Z
	晶闸管	V　V　阳极侧受控　阴极侧受控		限幅器	Z
W 传输通道，波导，天线	导线，电缆，母线	W			
	天线	W		均衡器	Z

1.3 电气图基本知识

电气图是根据国家电气制图标准，使用电气图例符号和文字符号以及规定的画法绘制而成的技术图纸。它包括电气控制系统图（电气原理图、电气接线图、电器元件布置图）、电气平面图、设备布局图、安装施工图、电气图例说明、设备材料明细表等。

（1）电气原理图

电气原理图表示电气控制线路的工作原理以及各电器元件的作用和相互关系，而不考虑各电器元件实际安装位置和实际连线情况，具有结构简单、层次分明、便于研究和电路分析等优点，如图 1–29 所示。

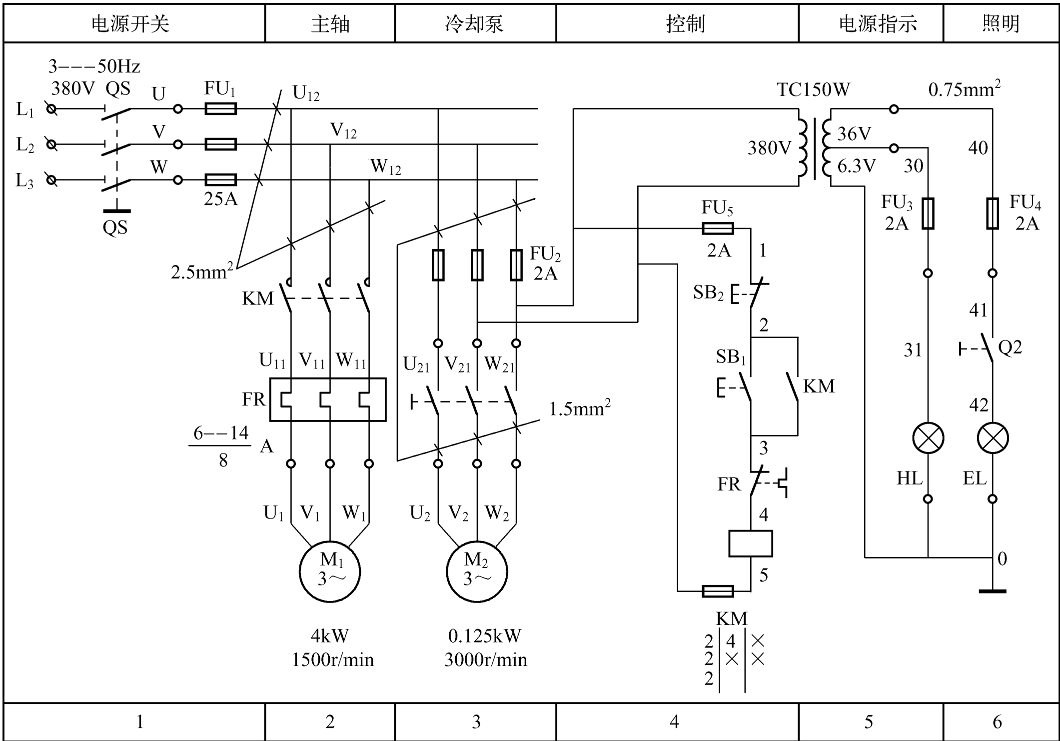


图 1–29 电气原理图

电气原理图根据控制对象的不同可分为主电路。主电路是将电源与电气设备（电动机或电负载）借助于低压电器进行可靠连接的电路，涉及的低压电器有低压断路器、熔断器、接触器（智能控制单元）、热过载保护器、接线端子等。控制电路是由主令电器、接触器和继电器的线圈、各种电器的常开和常闭辅助触点、电磁阀、电磁铁等按控制要求和控制逻辑进行的组合。

绘制电气原理图时，一般遵循下面的规则。

① 电气控制线路分主电路和控制电路。主电路用粗线绘出，而控制线路用细线画。一般主电路画在左侧，控制电路画在右侧。

② 电气控制线路中，同一电器的各导电部分如线圈和触点常常不画在一起，但要用同一文字符号标注。若有多个同类电器，可在文字符号后加上数字序号，如 KM1、KM2 等。

③ 电气控制线路的全部触点都按“非激励”状态绘出。“非激励”状态对电操作元件如接触器、继电器等是指线圈未通电时的触点状态；对机械操作元件如按钮、行程开关等是指没有受到外力时的触点状态；对主令控制器是指手柄置于“零位”时各触点的状态；断路器和隔离开关的触点处于断开状态。

④ 控制电路的分支线路，原则上按照动作先后顺序排列，两线交叉连接的电气连接点须用黑点标出，两线连接的接线端子用空心圆画出。

（2）电气接线图

电气接线图是将分布在电控柜和现场的电器元件和设备进行线路连接（如图 1–30 所示），绘制接线图时应把各电器的各个部分（如触点与线圈）画在一起，文字符号、元件连接顺序、线路号码编制必须与电气原理图一致。以安装接线为主，基本不涉及电气设备的整体结构和工作原理，着重表达接线过程。

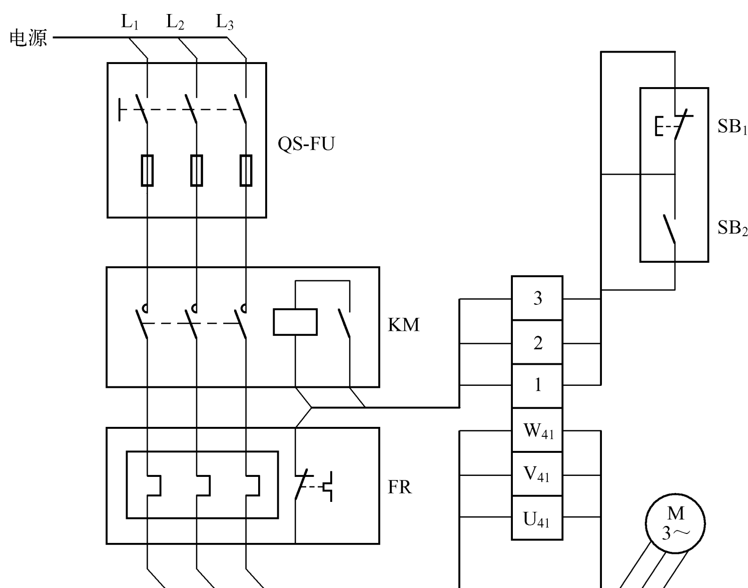


图 1–30　电气接线图

（3）电气元件布置图

电气元件布置图是器件的布局和位置安装（如图 1–31 所示），包括在电控柜和现场的分布，如电控柜中器件的分布、控制操作盘中器件的分布、器件的间隔和排放顺序、安装方式和定位等。在进行元器件布局时要注意整齐、美观、对称、外形尺寸与结构类型类似的电器安装在一起，以利于加工、安装和配线。在电气元件布置图中，一般标有各元件间距尺寸、安装孔距和进出线的方式。

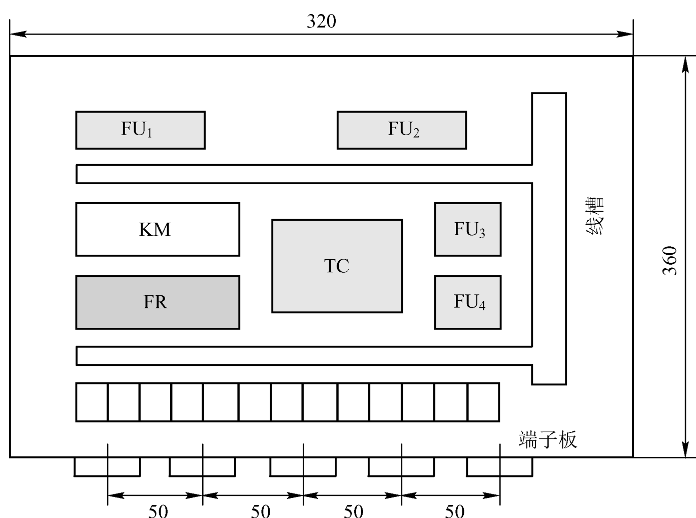


图 1-31　电气元件布置图

1.4　三相异步电动机的基本控制电路

1.4.1　基本控制环节

（1）自锁控制

自锁控制电路如图 1-32 所示。

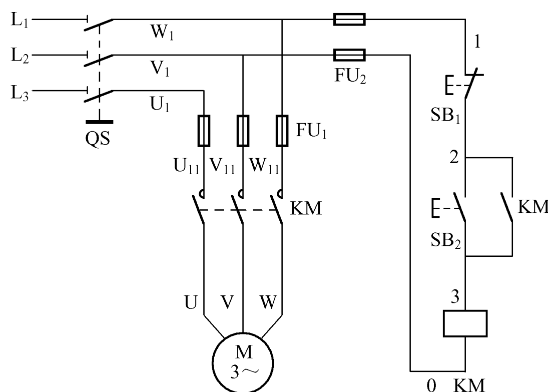


图 1-32　自锁控制

当按下 SB_2 启动按钮时，电流经 SB_1、SB_2 到达线圈 KM，接触器动作，接触器的主触点和辅助触点均闭合，电动机开始运转。松开 SB_2 时，电流经 SB_1、KM 的辅助触点到达线圈 KM，线圈保持一直得电。这种依靠接触器的辅助触点使线圈保持一直得电的方式称为自锁控制。当按下 SB_1 停止按钮时，线圈 KM 失电，所有触点返回，电动机停止转动。

这个电路是单向自锁控制电路，它的特点是，起动、保持、停止，所以称为“起、保、停”控制电路。

（2）点动控制

实际生产中，生产机械常需要进行点动控制，如机床调整对刀和刀架、立柱的快速移动等。所谓点动，指按下起动按钮，电动机转动；松开按钮，电动机停止运动。与之对应的，若松开按钮后能使电动机连续工作，则称为长动。区分点动与长动的关键是控制电路中控制电器通电后能否自锁，即是否具有自锁触点。控制电路如图 1–33 所示。

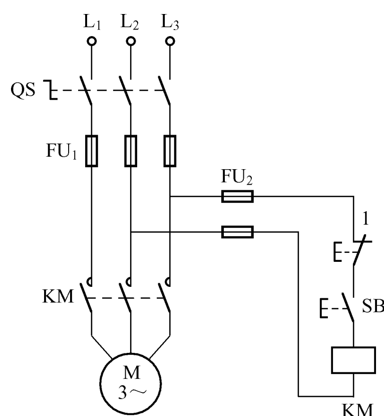


图 1–33　点动控制

生产实际中，有的生产机械既需要连续运转进行加工生产，又需要在进行调整工作时采用点动控制，这就产生了点动、长动混合控制电路。

（3）点动/长动混合控制

生产实际中，有的生产机械既需要连续运转进行加工生产，又需要在进行调整工作时采用点动控制，这就产生了点动、长动混合控制电路。常用控制方式如图 1–34 所示。

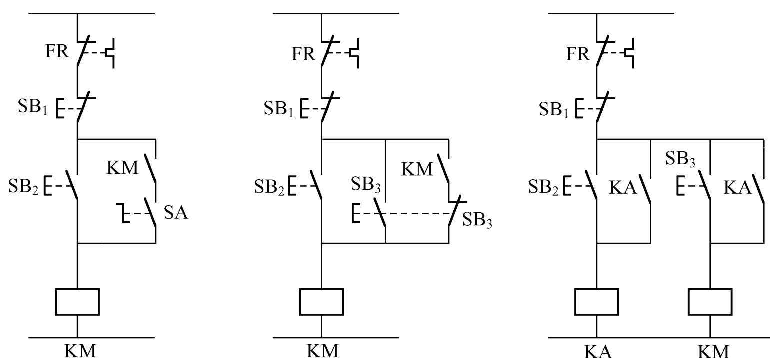


图 1–34　点动/长动混合控制

（4）多地点与多条件控制线路

多地点控制是指在两地或两个以上地点进行的控制操作，多用于规模较大的设备，为了操作方便常要求能在多个地点进行操作。在某些机械设备上，为保证操作安全，需要多个条件满足，设备才能工作。这样的控制要求可通过在电路中串联或并联电器的常闭触点和常开触点来实现。

多地点控制按钮的连接原则为：常开按钮均相互并联，组成“或”逻辑关系，常闭按钮均相互串联，组成“与”逻辑关系，任一条件满足，结果即可成立。遵循以上原则还可实现三地及更多地点的控制，电气控制线路如图 1–35a 所示。多条件控制按钮的连接原则为：常开按钮均相互串联，常闭按钮均相互并联，所有条件满足，结果才能成立，遵循以上原则还可实现更多条件的控制，电气控制线路如图 1–35b 所示。

（5）顺序控制线路

有多台电动机拖动的机械设备，在操作时为了保证设备的运行和工艺过程的顺利进行，对电动机的起动、停止，必须按一定顺序来控制，这就称为电动机的顺序控制，这种情况在机械设备中是常见的。例如，有的机床的油泵电动机要先于主轴电动机起动，主轴电动机又

先于切削液电动机起动等。电气控制线路如图 1-36 所示。

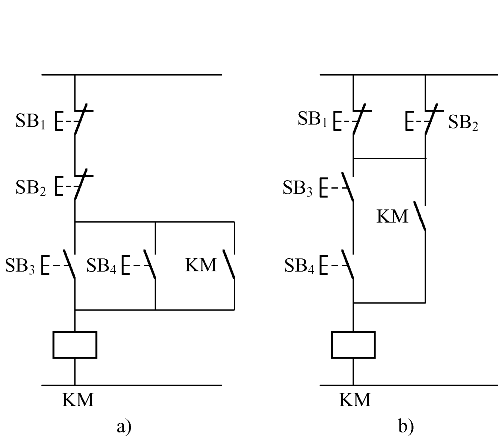


图 1-35　多地点与多条件控制线路

a）多地点控制线路　b）多条件控制线路

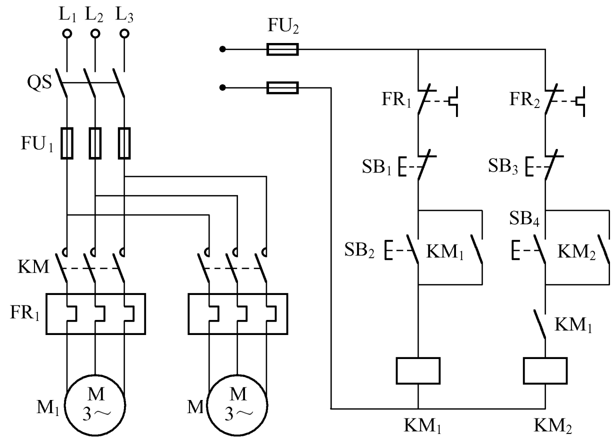


图 1-36　顺序控制线路

（6）正反转控制线路

生产实践中，许多设备均需要两个相反方向的运行控制，如机床工作台的进退、升降以及主轴的正反向运转等。此类控制均可通过电动机的正转与反转来实现。由电动机原理可知，电动机三相电源进线中任意两相对调，即可实现电动机的反向运转，电气控制线路如图 1-37 所示。

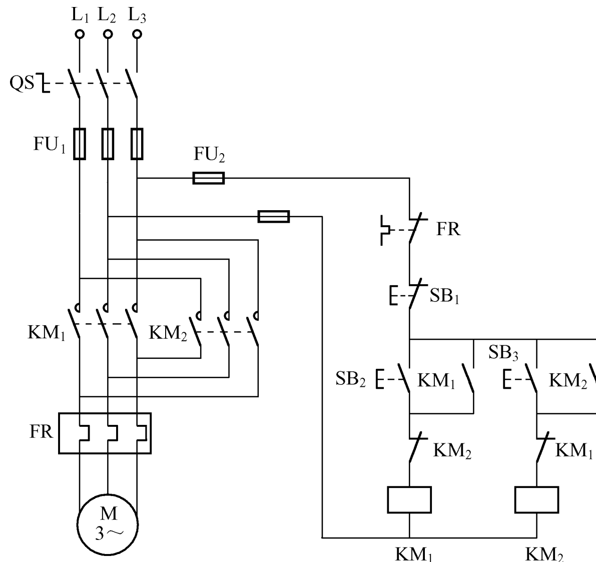


图 1-37　正反转控制线路

接触器 KM_1 和 KM_2 触点不能同时闭合，以免发生相间短路故障，因此需要在各自的控制电路中串接对方的常闭触点，构成互锁。

电动机由正转到反转，需先按停止按钮 SB_1，在操作上不方便，为了解决这个问题，可利用复合按钮进行控制，采用复合按钮，还可以起到联锁作用，这是由于按下 SB_2 时，只有 KM_1 可得电动作，同时 KM_2 回路被切断。同理按下 SB_3 时，只有 KM_2 可得电动作，同时

KM_1 回路被切断。电气控制线路如图 1–38 所示。

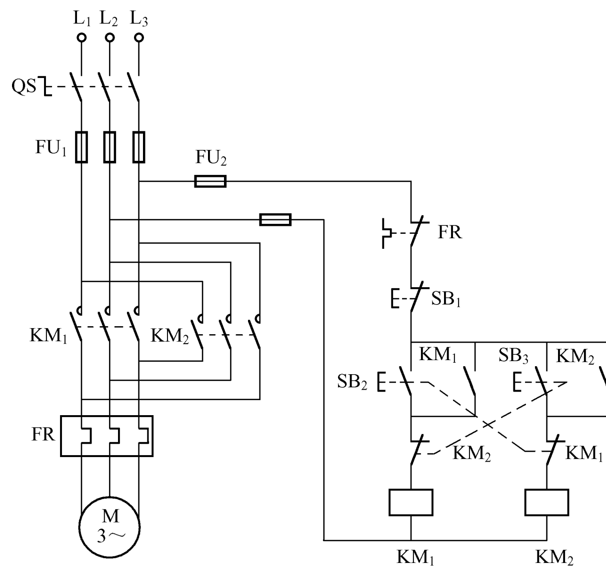


图 1–38　按钮联锁的正反转控制线路

但只用按钮进行联锁，而不用接触器常闭触点之间的联锁，是不可靠的。在实际中可能出现这样的情况，由于负载短路或大电流的长期作用，接触器的主触点被强烈的电弧“烧焊”在一起，或者接触器的机构失灵，使衔铁卡住总是在吸合状态。这都可能使主触点不能断开，这时如果另一接触器动作，就会造成电源短路事故。

如果用的是接触器常闭触点进行联锁，不论什么原因，只要一个接触器是吸合状态，它的联锁常闭触点就必然将另一接触器线圈电路切断，这就能避免事故的发生。

1.4.2　三相异步电动机起动控制

（1）笼型异步电动机直接起动控制线路

对容量较小，并且工作要求简单的电动机，如小型台钻、砂轮机、冷却泵的电动机，可用手动开关在动力电路中接通电源直接起动。

一般中小型机床的主电动机采用接触器直接起动，接触器直接起动电路分为两部分，主电路由接触器的主触点接通与断开，控制电路由按钮和辅助常开触点控制接触器线圈的通断电，实现对主电路的通断控制。电气控制线路如图 1–39 所示。

直接启动的优点是电气设备少，线路简单。实际的直接启动电路一般采用空气开关直接启动控制。对于容量大的电动机来说，由于启动电流大，会引起较大的电网压降，所以必须采用减压启动的方法，以限制启动电流。

（2）笼型异步电动机降压起动控制线路

容量大于 10 kW 的笼型异步电动机直接起动时，起动冲击电流为额定值的 4 ~ 7 倍，故一般均需采取相应措施降低电压，即减小与电压成正比的电枢电流，从而在电路中不至于产生过大的电压降。常用的降压起动方式有定子电路串电阻降压起动、星形 – 三角形（Y – △）降压起动和自耦变压器降压起动。

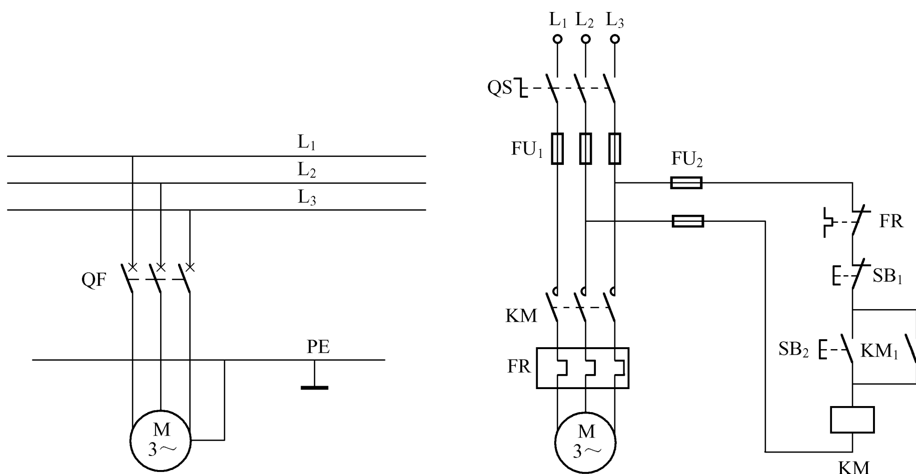


图 1–39　笼型异步电动机直接起动控制线路

① 星形 – 三角形降压起动控制电路

正常运行时，定子绕组为三角形联结的笼型异步电动机，可采用星形 – 三角形的降压起动方式来达到限制起动电流的目的。起动时，定子绕组首先联结成星形，待转速上升到接近额定转速时，将定子绕组的联结由星形联结成三角形，电动机便进入全压正常运行状态。电气控制线路如图 1–40 所示。

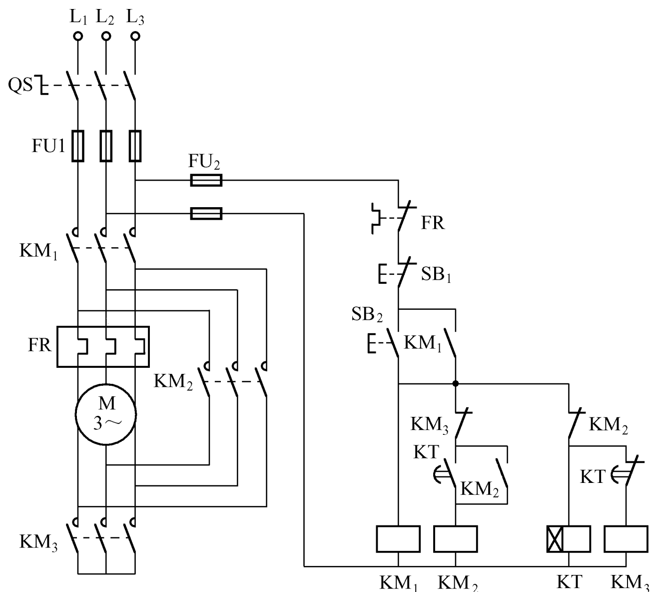


图 1–40　星形 – 三角形降压起动控制

② 定子串电阻降压起动控制电路

电动机串电阻降压起动是电动机起动时，在三相定子绕组中串接电阻分压，使定子绕组上的压降降低，起动后再将电阻短接，电动机即可在全压下运行。这种起动方式不受接线方式的限制，设备简单，常用于中小型设备和用于限制机床点动调整时的起动电流。电气控制线路如图 1–41 所示。

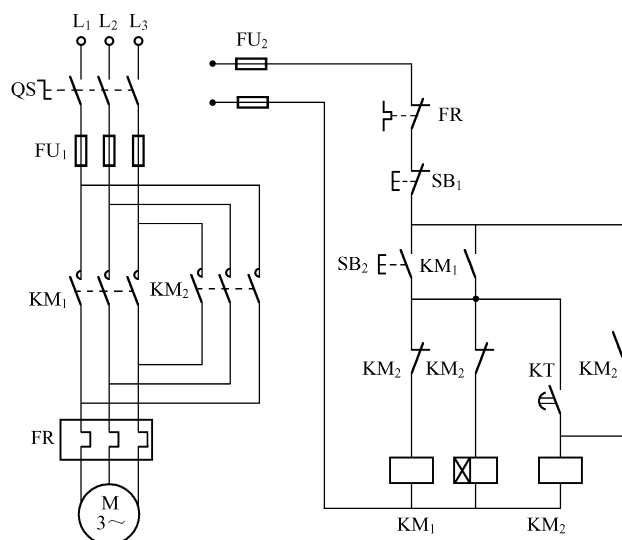


图 1–41　定子串电阻降压起动控制电路

③ 自耦变压器降压起动控制电路

在自耦变压器降压起动的控制线路中，电动机起动电流的限制，是依靠自耦变压器的降压作用来实现的。电动机起动的时候，定子绕组得到的电压是自耦变压器的二次电压。一旦起动结束，自耦变压器便被切除，额定电压通过接触器直接加于定子绕组，电动机进入全压运行的正常工作。电气控制线路如图 1–42 所示。

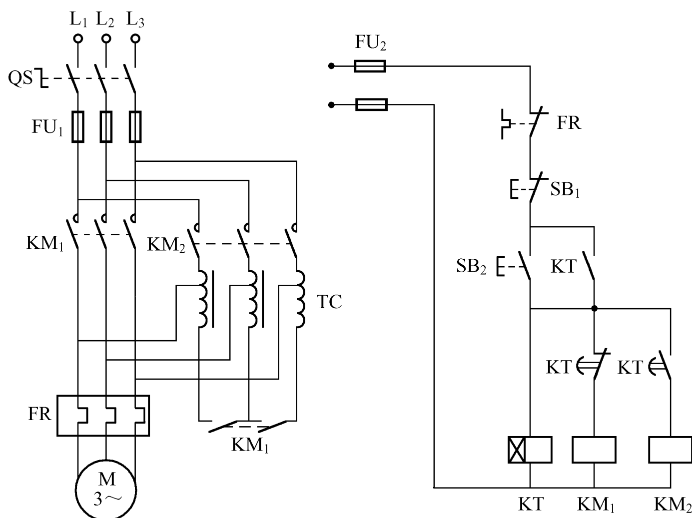


图 1–42　自耦变压器降压起动控制电路

1. 4. 3　三相异步电动机制动控制

三相异步电动机从切除电源到完全停止运转。由于惯性的关系，总要经过一段时间，这往往不能适应某些生产机械工艺的要求。如万能铣床、卧式镗床、电梯等，为提高生产效率

及准确停位，要求电动机能迅速停车，对电动机进行制动控制。制动方法一般有两大类：机械制动和电气制动。电气制动中常用反接制动和能耗制动。

（1）反接制动控制线路

反接制动控制的工作原理：改变异步电动机定子绕组中的三相电源相序，使定子绕组产生方向相反的旋转磁场，从而产生制动转矩，实现制动。反接制动要求在电动机转速接近零时及时切断反相序的电源，以防止电动机反向起动。

反接制动过程为：当想要停车时，首先将三相电源切换，然后当电动机转速接近零时，再将三相电源切除。电气控制线路如图 1–43 所示。

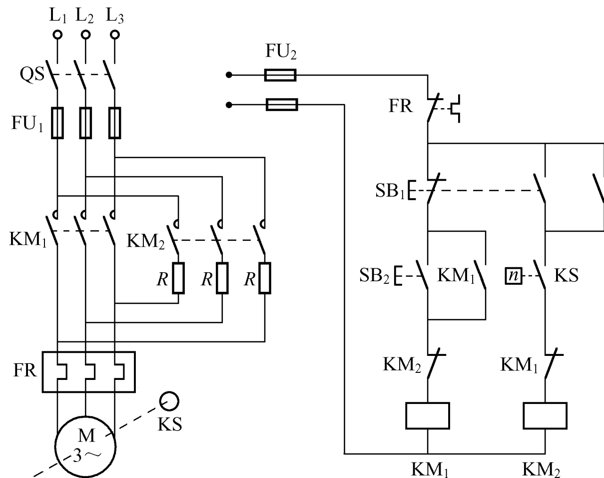


图 1–43　反接制动控制线路

控制线路中停止按钮使用了复合按钮 SB_1，并在其常开触点上并联了 KM_2 的常开触点，使 KM_2 能自锁。这样在用手转动电动机时，虽然 KS 的常开触点闭合，但只要不按复合按钮 SB_1，KM_2 就不会通电，电动机也就不会反接于电源，只有按下 SB_1，KM_2 才能通电，制动电路才能接通。因电动机反接制动电流很大，故在主回路中串入电阻 R，可防止制动时电动机绕组过热。

（2）能耗制动控制线路

能耗制动控制的工作原理：在三相电动机停车切断三相交流电源的同时，将一直流电源引入定子绕组，产生静止磁场。电动机转子由于惯性仍沿原方向转动，则转子在静止磁场中切割磁力线，产生一个与惯性转动方向相反的电磁转矩，实现对转子的制动。电气控制线路如图 1–44 所示。

反接制动时，制动电流很大，因此制动力矩大，制动效果显著，但在制动时有冲击，制动不平稳且能量消耗大。能耗制动与反接制动相比，制动平稳，准确，能量消耗少，但制动力矩较弱，特别在低速时制动效果差，并且还需提供直流电源。在实际使用时，应根据设备的工作要求选用合适的制动方法。

1.4.4　三相异步电机调速控制线路

实际生产中，对机械设备常有多种速度输出的要求，通常采用单速电动机时，需配有机

械变速系统以满足变速要求。当设备的结构尺寸受到限制或要求速度连续可调时，常采用多速电动机或电动机调速。

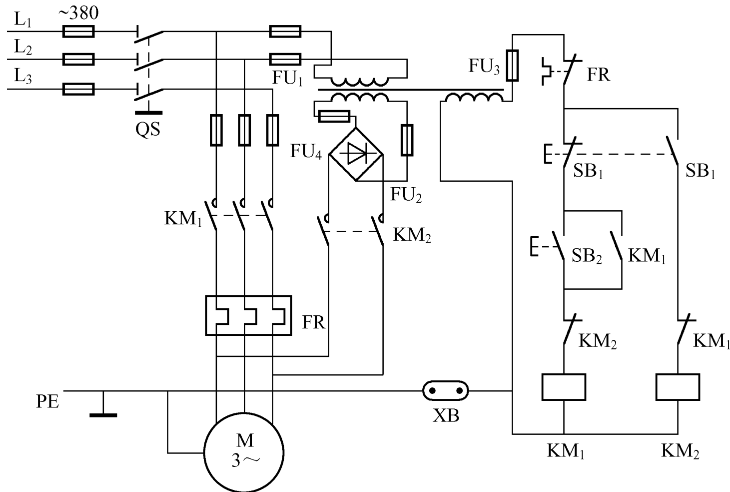


图 1-44　能耗制动控制线路

根据三相异步电动机的转速公式：

$$n = \frac{60f_1}{p}(1-s)$$

得出三相异步电动机的调速可使用改变电动机定子绕组的磁极对数，改变电源频率或改变转差率的方式。

三相笼型电动机采用改变磁极对数调速。当改变定子极数时，转子极数也同时改变。笼型转子本身没有固定的极数，它的极数随定子极数而定。电动机变极调速的优点是，它既适用于恒功率负载，又适用于恒转矩负载，线路简单，维修方便；缺点是有级调速且价格昂贵。

改变定子绕组极对数的方法如下。

1）装一套定子绕组，改变它的连接方式，得到不同的极对数。

2）定子槽里装两套极对数不一样的独立绕组。

3）定子槽里装两套极对数不一样的独立绕组，而每套绕组本身又可以改变它的连接方式，得到不同的极对数，如图 1-45 所示。

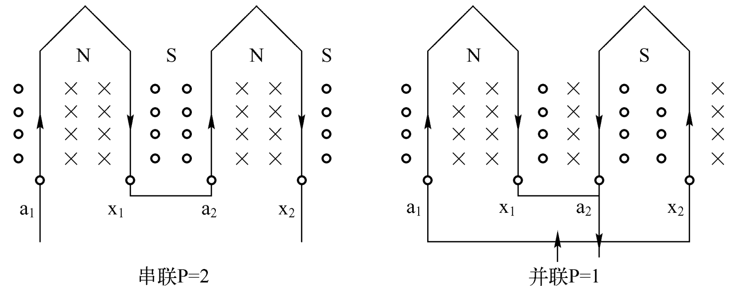


图 1-45　改变定子绕组极对数

调速控制线路如图 1-46 所示。

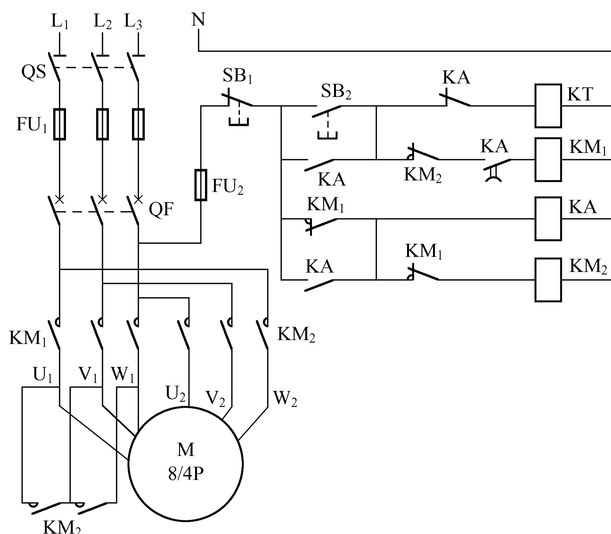


图 1-46　调速控制线路

习题

1-1　常用的主令电器有哪些？它们的主要用途是什么？

1-2　选用接触器时应注意哪些问题？接触器和中间继电器有何差异？

1-3　热继电器在电路中可以起短路保护吗？为什么？

1-4　既然在电动机的主电路中装有熔断器，为什么还要装热继电器？装有热继电器是否就可以不装熔断器？为什么？

1-5　行程开关、万能转换开关及主令控制器在电路中各起什么作用？

1-6　试设计对一台电动机可以进行两处操作的长动和点动控制线路。

1-7　在电动机直接启动电路中，按下启动按钮后电机不转可能的原因都有哪些？

1-8　什么叫互锁？如何实现 ？

1-9　电动机为什么要进行制动？

1-10　三相异步电动机调速的方法都有哪些？

第 2 章　可编程序控制器概述

2.1　PLC 简介

可编程控制器（Programmable Controller，PC）是新一代的工业控制装置，是工业自动化的基础平台。目前已被广泛应用到石油、化工、电力、机械制造、汽车、交通等各个领域。早期的可编程控制器只能用于进行逻辑控制，因此被称为可编程逻辑控制器（Programmable Logic Controller，PLC）。随着现代技术的发展，可编程控制器用微处理器作为其控制的核心部件，其控制的功能也远远超过了逻辑控制的范围，于是这种装置被称为可编程控制器（Programmable Controller）简称为 PC。但是为了避免与个人计算机（Personal Computer，PC）相混淆，可编程控制器仍然被简称为 PLC。

2.1.1　PLC 的产生

PLC 产生之前，继电器控制系统广泛应用于工业生产的各个领域，起着不可替代的作用。随着生产规模的逐步扩大，继电器控制系统已越来越难以适应现代工业生产的要求。继电器控制系统通常是针对某一固定的动作顺序或生产工艺而设计的，它的控制功能仅局限于逻辑控制、定时、计数等一些简单的控制，一旦动作顺序或生产工艺发生变化，就必须重新进行设计、布线、装配和调试，造成时间和资金的严重浪费。另外继电器控制系统体积大、耗电多、可靠性差、寿命短、运行速度慢、适应性差。为了改变这一现状，1968 年美国最大的汽车制造商通用汽车公司（GM），为了适应汽车型号不断更新的需求，并能在竞争激烈的汽车工业中占有优势，提出要研制一种新型的工业控制装置来取代继电器控制装置。为此，拟定了 10 项公开招标的技术要求（GM10 条）。

1）编程简单，可在现场修改程序。

2）维护方便，采用插件式结构。

3）可靠性高于继电器控制系统。

4）体积小于继电器控制柜。

5）可将数据直接送入管理计算机。

6）在成本上可与继电器控制柜竞争。

7）输入可以是交流 115 V。

8）输出是交流 115 V，2 A 以上，可直接驱动电磁阀等。

9）在扩展时，原系统只要很小变更。

10）用户程序存储器容量至少能扩展到 4 KB。

根据这些要求，1969 年美国数字设备公司（DEC）研制出了世界上第一台 PLC，并在美国通用汽车公司自动装配生产线上试用成功。这种新型的工控装置，以其体积小、可靠性高、使用寿命长、简单易懂、操作维护方便等一系列优点，很快就在美国许多行业里得到推

广和应用，同时也受到了世界上许多国家的高度重视。1971 年，日本从美国引进了这项新技术，并研制出了日本第一台 PLC。1973 年西欧一些国家也研制出了自己的 PLC。我国从 20 世纪 70 年代中期开始研制 PLC，1977 年我国采用美国 Motorola 公司的集成芯片研制成功了国内第一台有实用价值的 PLC。

2.1.2 PLC 的发展

可编程控制器（PLC）自问世以后就凭借其优越的性能得到了迅速的发展，现在 PLC 已经成为一种最重要的也是应用场合最多的工业控制器。

最初的 PLC 限于当时元器件的条件及计算机的发展水平，主要由分立元件和中小规模集成电路组成，存储器采用的是磁芯存储器。它只能完成简单的开关量逻辑控制以及定时、计数功能。这时的 PLC 主要是被用作继电器控制装置的替代品，但它的性能要优于继电器，其主要优点包括体积小、易于安装、能耗低、简单易学等。为了方便熟悉继电器、接触器系统的工程技术人员使用，可编程控制器在软件编程上采用和继电器控制电路相似的梯形图作为主要的编程语言。

20 世纪 70 年代出现的微处理器使可编程控制器发生了巨大的变化。欧美及日本的一些厂家以微处理器和大规模集成电路芯片作为 PLC 的中央处理单元（CPU），使 PLC 增加了运算、数据传送及处理通信、自诊断等功能，可靠性也得到了进一步的提升。PLC 成为了真正具有计算机特征的工业控制装置。70 年代中后期，可编程控制器进入实用化发展阶段，计算机技术已全面引入可编程控制器中，使其功能发生了飞跃。更高的运算速度、更小的体积、更可靠的工业抗干扰设计、模拟量运算、PID 功能以及极高的性价比奠定了 PLC 在现代工业中的地位。

20 世纪 80 年代至 90 年代中期，可编程控制器在先进工业国家中已获得广泛应用。这个时期可编程控制器发展的特点是大规模、高速度、高性能、产品系列化。PLC 在处理模拟量能力、数字运算能力、人机接口能力和网络能力等方面得到大幅度提高，PLC 逐渐进入过程控制领域，在某些应用上取代了在过程控制领域处于统治地位的 DCS 系统。这个时期 PLC 的另一个特点是世界上生产可编程控制器的国家日益增多，产量日益上升。这标志着可编程控制器已步入成熟阶段。

20 世纪 90 年代末期至今，可编程控制器的发展更加适应于现代工业的需要。从产品规模上来看，PLC 会进一步向超小型及超大型方向发展；从控制能力上来看，诞生了各种各样的特殊功能单元，用于压力、温度、转速、位移等各式各样的控制场合；从产品的配套能力来看，生产了各种人机界面单元、通信单元，使应用可编程控制器的工业控制设备的配套更加容易。目前，可编程控制器在机械制造、石油化工、冶金钢铁、汽车、轻工业等领域的应用都得到了长足的发展。伴随着计算机网络的发展，可编程控制器作为自动化控制网络和国际通用网络的重要组成部分，将在工业及工业以外的众多领域发挥越来越大的作用。

2.1.3 PLC 的定义

1987 年国际电工委员会（International Electrotechnical Commission，IEC）在可编程控制器国际标准草案中对可编程控制器做出如下定义：可编程控制器是一种数字运算操作的电子系统，专为在工业环境下应用而设计。它采用了可编程序的存储器，用于在其内部存储执行

逻辑运算、顺序控制、定时、计数和算术运算等操作的指令，并通过数字式、模拟式的输入和输出，控制各种类型的机械或生产过程。可编程控制器及其有关的外围设备都应按易于与工业控制系统形成一个整体、易于扩充其功能的原则设计。

由 PLC 的定义可以看出，PLC 具有和计算机相类似的结构，也是一种工业通用计算机，只不过 PLC 为适应各种较为恶劣的工业环境而设计，具有很强的抗干扰能力，这也是 PLC 区别于一般微机控制系统的一个重要特征，并且 PLC 必须经过用户二次开发编程才能使用。

2.1.4　PLC 的分类

PLC 是根据现代化大生产的需要而产生的，PLC 的分类也必然要符合现代化生产的需求。PLC 产品的种类繁多，其功能、内存容量、控制规模、外形等方面均存在较大差异，型号规格不统一，还没有一个权威的统一分类标准，准确分类也是困难的。目前，一般按照控制规模、结构形式和实现的功能粗略地对 PLC 进行分类。

（1）按 PLC 的控制规模分类

控制规模主要指 PLC 可控制的最大 I/O 点数。通常而言，PLC 能控制的 I/O 点数越多，其控制的对象就越复杂，控制系统的规模也越大。PLC 按控制规模分，可以分为小型机、中型机和大型机 3 类。

① 小型机

小型机的控制点数一般在 256 点以内，通常采用整体式结构，适用于机电一体化设备或各种自动化仪表的单机控制。如日本欧姆龙公司生产的 CQM1、三菱公司生产的 FX2 和德国西门子公司生产的 S7－200。这类 PLC 由于控制点数不多，控制功能有一定局限性。但它价格低廉，并且小巧、灵活，可以直接安装在电气控制柜内，很适合用于单机控制或小型系统的控制。

② 中型机

中型机的控制点数一般在 256～2048 点之间，一般采用模块式结构，常用于大型机电一体化设备的控制。如日本欧姆龙公司生产的 C200H、富士公司生产的 HDC－100 和德国西门子公司生产的 S7－300。这类 PLC 由于控制点数较多，控制功能较强，有些 PLC 还有较强的计算能力，不仅可用于对设备进行直接控制，也可以对多个下一级的 PLC 进行监控，适用于中型或大型控制系统的控制。

③ 大型机

大型机的控制点数一般大于 2048 点，大型 PLC 使用 32 位微处理器，多 CPU 并行工作，并具有大容量存储器。均采用模块式结构，具有较强的网络通信功能，可用于大型自动化生产过程，组成分布式控制系统。如日本欧姆龙公司生产的 C2000H、富士公司生产的 F200 和德国西门子公司生产的 S7－400。这类 PLC 控制点数多，控制功能很强，有很强的计算能力。同时，这类 PLC 运行速度很高，不仅能完成较复杂的算术运算，还能进行复杂的矩阵运算，它不仅可以用于对设备进行直接控制，可以对多个下一级的 PLC 进行监控，还可以完成现代化工厂的全面管理和控制任务。

上述划分方式并不十分严格，也不是一成不变的。

（2）按 PLC 的结构分类

为了方便在工业现场安装，便于扩展，方便接线，其结构与普通计算机有很大区别。通

常从组成结构形式上将 PLC 分为整体式和模块式两大类。

① 整体式

整体式结构的 PLC 把电源、CPU、存储器和 I/O 系统都集成在一个单元内，该单元叫作基本单元。一个基本单元就是一台完整的 PLC。控制点数不满足需要时，可再接扩展单元，扩展单元不带 CPU，在安装时不用基板，仅用电缆进行单元间的连接，由基本单元和若干扩展单元组成较大的系统。整体式结构的特点是紧凑、体积小、成本低、安装方便，其缺点是各个单元输入与输出点数有确定的比例，使 PLC 的配置缺少灵活性，有些 I/O 资源不能充分利用。早期的小型机多为整体式结构。

② 模块式

PLC 的模块式结构通常也叫作组合式结构。模块式结构的 PLC 是把 PLC 系统的各个组成部分按功能分成若干个模块，如 CPU 模块、输入模块、输出模块和电源模块等，其中各模块功能比较单一，模块的种类却日趋丰富。例如，一些 PLC 除了一些基本的 I/O 模块外，还有一些特殊功能模块，如温度检测模块、位置检测模块、PID 控制模块和通信模块等。模块式结构的 PLC 采用搭积木的方式，在一块基板插槽上插上所需模块组成控制系统（又叫作组合式结构）。有的 PLC 没有基板而是采用电缆把模块连接起来组成控制系统（又叫作叠装式结构）。模块式结构的 PLC 特点是 CPU、输入和输出均为独立的模块。模块尺寸统一、安装整齐、I/O 点选型自由，并且安装调试、扩展和维修方便。中型机和大型机多为模块式结构。

（3）按 PLC 的功能分类

PLC 按功能强弱来分，可以分为低档机、中档机和高档机 3 类。

① 低档机

低档机具有基本的控制功能和一般的运算能力。工作速度比较低，能带的输入/输出模块的数量比较少，种类也比较少。这类可编程序控制器只适合于小规模的简单控制，在联网中一般适合做从站使用。如日本欧姆龙公司生产的 C60P 就属于低档机。

② 中档机

中档机具有较强的控制功能和较强的运算能力，它不仅能完成一般的逻辑运算，也能完成比较复杂的三角函数、指数运算和 PID 运算。工作速度比较快，能带的输入/输出模块的数量和种类也比较多。这类可编程序控制器不仅能完成小型系统的控制，也可以完成较大规模的控制任务。在联网中可以做从站，也可以做主站。如德国西门子公司生产的 S7－300 就属于中档机。

③ 高档机

高档机具有强大的控制功能和强大的运算能力，它不仅能完成逻辑运算、三角函数运算、指数运算和 PID 运算，还能进行复杂的矩阵运算。工作速度很快，能配带的输入/输出模块的数量很多，种类也很全面。这类可编程序控制器不仅能完成中等规模的控制工程，也可以完成规模很大的控制任务。在联网中一般做主站使用。如德国西门子公司生产的 S7－400 就属于高档机。

2.1.5 PLC 的特点

PLC 具有通用性强、使用方便、适应面广、可靠性高、抗干扰能力强、编程简单等优越

的性能，这些特点使其在工业自动化控制特别是顺序控制领域拥有无法取代的地位。

（1）可靠性高、抗干扰能力强

在传统的继电器控制系统中，由于器件的老化、脱焊、触点的抖动、触点的电弧、接触不良等现象，大大降低了系统的可靠性。继电器控制系统的维修不仅耗费时间金钱，重要的是由于维修停产所带来的经济损失更是不可估量的。在 PLC 控制系统中，由于大量的开关动作是由无触点的半导体电路完成的，而且 PLC 在硬件和软件方面都采取了强有力的措施，使得产品具有极高的可靠性和抗干扰性。

在硬件方面 PLC 对所有的 I/O 接口电路都采用光电隔离措施，使工业现场的外部电路与 PLC 内部电路之间被有效地隔离开来，以减少故障和误动作；电源、CPU、编程器等都采用屏蔽措施，防止外界的干扰；供电系统及输入电路采用多种形式的滤波，以消除或抑制高频干扰，也削弱了各个部分之间的相互影响；采用模块式结构，当某一模块出现故障时，可以迅速更换该模块，从而尽可能缩短系统的故障停机时间。

在软件方面，PLC 设置了监视定时器，如果程序每次循环的执行时间超过了设定值，则表明程序已经进入死循环，可以立即报警。PLC 具有良好的自诊断功能，一旦电源或其他软件、硬件发生异常情况，CPU 会立即把当前状态保存起来，并禁止对程序的任何操作，以防止存储信息被冲掉，等故障排除后则会立即恢复到故障前的状态继续执行程序。另外 PLC 还加强对程序的检查和校验，当发现错误时会立即报警，并停止程序的执行。

（2）编程方法简单易学

大多数的 PLC 采用梯形图语言编程，其电路符号和表达方式与继电器电路原理图相似。只用少量的开关量逻辑控制指令就可以很方便地实现继电器电路的功能。另外梯形图语言形象直观、编程方便、简单易学，熟悉继电器控制电路图的电器技术人员很快就可以熟悉梯形图语言，并用于进行编写程序。

（3）灵活性和通用性强

PLC 是利用程序来实现各种控制功能的。在 PLC 控制系统中，当控制功能改变时只需修改控制程序即可，PLC 的外部接线一般只需做少许改动。一台 PLC 可以用于不同的控制系统，只要加载相应的程序就行了。而继电器控制系统中当工艺要求稍有改变时，控制电路就必须随之做相应的变动，耗时又费力。所以说 PLC 的灵活性和通用性是继电器电路所无法比拟的。

（4）丰富的 I/O 接口模块

PLC 针对不同的工业现场信号，如交流或直流、开关量或模拟量、电压或电流、脉冲或电位、强电或弱电信号，都能选择到相应的 I/O 模块与之相匹配。对于工业现场的器件或设备，如按钮、行程开关、接近开关、传感器及变送器、电磁线圈、控制阀等设备都有相应的 I/O 模块与之相连接。另外，为了提高 PLC 的操作性能，还有多种人机对话的接口模块；为了组成工业局域网络，还有多种通信联网的接口模块。

（5）采用模块化结构

为了适应各种工业控制的需求，除了单元式的小型 PLC 以外，绝大多数 PLC 都采用模块化结构。PLC 的各个部件，包括 CPU、电源、I/O 接口等均采用模块化结构，并由机架及电缆将各模块连接起来。系统的规模和功能可以根据用户自己的需要自行组合。

（6）易于设计及调试

由于 PLC 是通过程序来实现对系统的控制的，所以设计人员可以在实验室里设计和修改程序。还可以在实验室里进行系统的模拟运行和调试，使工作量大大减少。而继电器控制系统是靠调整控制电路的接线来改变其控制功能的，调试时费时又费力。

（7）体积小、能耗低、易于实现机电一体化

小型 PLC 的体积仅相当于几个继电器的大小，其内部电路主要采用半导体集成电路，具有结构紧凑、体积小、重量轻、功耗低的特点。PLC 还具有很强的抗干扰能力，能适应各种恶劣的环境，因此 PLC 是实现机电一体化的理想控制装置。

2. 1. 6　PLC 性能指标

PLC 的主要性能指标包括以下几个方面。

（1）输入/输出点数

I/O 点数即 PLC 面板上的输入、输出端子的个数，这是一项重要的技术指标。I/O 点数越多，外部可接的输入器件和输出器件就越多，控制规模也就越大。通常小型机最多有几十个点，中型机有几百个点，大型机超过千点。

（2）存储容量

PLC 中的存储器包括系统存储器和用户程序存储器。这里的存储容量是指用户程序存储器的容量。用户程序存储器容量越大，可存储的程序就越大，可以控制的系统规模也就越大。一般以字节（B）为单位。

（3）扫描速度

扫描速度是指 PLC 执行程序的速度，是衡量 PLC 性能的重要指标之一，主要取决于所用芯片的性能。一般以执行 1000 步指令所需的时间来衡量，单位为 ms/千步。有时也以执行 1 步指令的时间计算，单位为 μs/步。扫描速度越快，PLC 的响应速度也越快，对系统的控制也就越及时、准确、可靠。

（4）指令的数量和功能

用户编写的程序所完成的控制任务，取决于 PLC 指令的多少。编程指令的数量和功能越多，PLC 的处理能力和控制能力就越强。

（5）内部器件的种类和数量

内部器件包括各种继电器、计数器、定时器、数据存储器等。其种类和数量越多，存储各种信息的能力和控制能力就越强。

（6）可扩展性

在选择 PLC 时，需要考虑其可扩展性。它主要包括输入、输出点数的扩展，存储容量的扩展，联网功能的扩展和可扩展模块的多少。

2. 2　PLC 硬件组成

PLC 本身就是一台适合工业现场使用的专用计算机，其硬件结构如图 2-1 所示。

PLC 是一种以微处理器为核心的专用于工业控制的特殊计算机，其硬件组成与一般的微型计算机相类似，虽然不同厂家 PLC 的结构多种多样，但其基本结构是相同的，即主要由

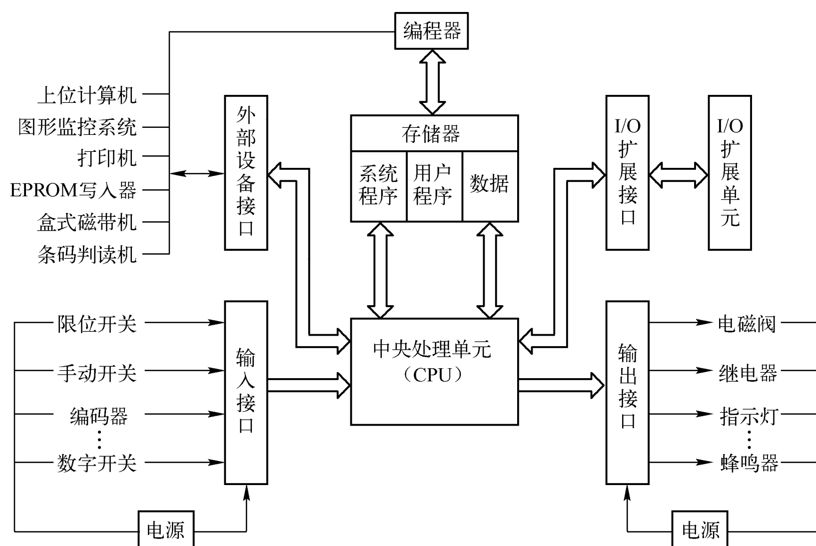


图 2-1　PLC 硬件结构图

中央处理器（CPU）、存储器、输入/输出单元、电源、I/O 扩展端口、通信单元等有机组合而成。根据结构的不同，PLC 可以分为整体式和组合式（也称模块式）两类。整体式 PLC 所有部件都装在同一机壳内，结构紧凑、体积小。小型机常采用这种结构，如德国西门子（SIEMENS）公司的 S7-200 系列 PLC。组合式 PLC 是将组成 PLC 的多个单元分别做成相应的模块，各模块在导轨上通过总线连接起来。大中型 PLC 常采用这种方式，如西门子公司的 S7-300/400 系列 PLC。西门子公司整体式 PLC 如图 2-2 所示，组合式 PLC 如图 2-3 所示。



图 2-2　整体式 PLC

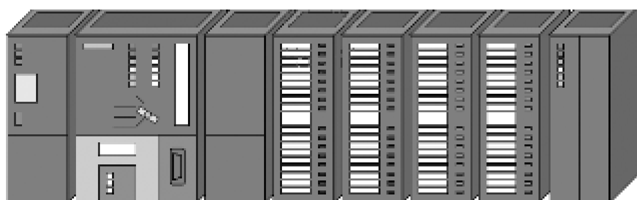


图 2-3　组合式 PLC

（1）中央处理器单元（CPU）

CPU 是 PLC 的核心部件，能使 PLC 按照预先编好的系统程序来完成各种控制。小型 PLC 多用 8 位微处理器或单片机；中型 PLC 多用 16 位微处理器或单片机；大型 PLC 多用双极型位片机。其作用如下。

① 接收并存储用户程序和数据。

② 接收、调用现场输入设备的状态和数据。先将现场输入的数据保存起来，在需要用的时候调用该数据。

③ 诊断电源及 PLC 内部电路的工作状态和编程过程中的语法错误，发现错误时会立即报警。

④ 当 PLC 进入运行（Run）状态时，CPU 根据用户程序存储的先后顺序依次执行，完成程序中规定的操作。

⑤ 根据程序运行的结果更新有关标志位的状态和输出映像寄存器的内容，再经输出部件实现输出控制或数据通信功能。

（2）存储器

PLC 的存储器是用于存储数据和程序的，可以分为系统程序存储器（ROM 或 EPROM）、用户程序存储器（RAM）、工作数据存储器（RAM/FLASH）。系统程序存储器决定了 PLC 的功能，它是只读存储器，用户不能更改其内容。PLC 中常用 RAM 来存储用户程序，RAM 的工作速度快，价格便宜、改写方便，同时在 PLC 中配有锂电池，当外部电源断电时，可以保存 RAM 中的信息。用于存储工作数据的区域称为工作数据区。工作数据是经常变化和存取的，所以工作数据存储器必须是可读写的。

（3）输入/输出单元

输入/输出单元是 PLC 与外部设备互相联系的窗口。实际的生产中信号电平是多样的，外部执行机构所需要的电平也是不同的。但是 CPU 所处理的信号只能是标准电平，因此需要通过输入/输出单元来实现对这些信号电平的转换。它实质上是 PLC 与被控对象之间传送信号的接口部件。输入单元接收现场设备向 PLC 提供的信号，如按钮、开关、继电器触点、拨码器等开关量信号。这些信号经过输入电路的滤波、光电隔离、电平转换等处理后变成 CPU 能够接收和处理的信号。输出单元将经过 CPU 处理的微弱电信号通过光电隔离、功率放大等处理后转换成外部设备所需要的强电信号，从而来驱动各种执行元件，如接触器、电磁阀、调节器、调速装置等。

（4）电源

一般情况下 PLC 使用 220 V 的交流电源或 24 V 的直流电源。电源部件将外部输入的交流电经整流滤波处理后转换成供 PLC 的中央处理器、存储器等内部电路工作所需要的 5 V、12 V、24 V 等不同电压等级的直流电源，使 PLC 能正常工作。许多 PLC 的直流电源多采用直流开关稳压电源，不仅可以提供多路独立的电压供内部电路使用，还可以向外部提供 24 V 的直流电源，给输入单元所连接的外部开关或传感器供电。

一般对于整体式 PLC，电源部件封装在主机内部，对于模块式 PLC，电源部件一般采用单独的电源模块。

（5）I/O 扩展端口

PLC 的 I/O 端口是十分重要的资源，扩展 I/O 端口是提高 PLC 控制系统经济性能指标的重要手段。当 PLC 主控单元的 I/O 点数不能满足用户的需求时，可以通过 I/O 扩展端口用扁平电缆将 I/O 扩展单元与主控单元相连，以增加 I/O 点数。大部分的 PLC 都有扩展端口。主机可以通过扩展端口连接 I/O 扩展单元来增加 I/O 点数，也可以通过扩展端口连接各种特殊功能单元以扩展 PLC 的功能。

（6）外设端口

PLC 可以通过外设端口与各种外部设备相连接。例如连接终端设备进行程序的设计、调试和系统监控；连接打印机可以打印用户程序、打印 PLC 运行过程中的状态、打印故障报警的种类和时间等；连接 EPROM 写入器，将调试好的用户程序写入 EPROM，以免被误改动等；有的 PLC 还可以通过外部设备端口与其他 PLC、上位机进行通信或加入各种网络。

（7）编程工具

编程工具是开发应用和检查维护 PLC 以及监控系统运行不可或缺的外部设备。利用编程工具可以将用户程序输入到 PLC 的存储器，还可以检查、修改、调试程序以及监视程序的运行。PLC 的编程工具有两种形式：一种是手持编程器，它由键盘、显示器和工作方式选择开关等组成，主要用于调试简单的程序、现场修改参数以及监视 PLC 自身的工作情况；另一种是利用上位计算机中的专业编程软件（如西门子 S7－300PLC 用的 STEP 7 软件），它主要用于编写较大型的程序，并能够灵活地修改、下载、安装程序以及在线调试和监控程序。编程软件的应用更为广泛。

（8）智能单元

智能单元是 PLC 中的一个模块，它与 CPU 通过系统总线连接，并在 CPU 的协调管理下独立地进行工作。常用的智能单元包括高速计数器单元、A/D 单元、D/A 单元、位置控制单元、PID 控制单元、温度控制单元等。

2.3　PLC 编程环境和工作原理

2.3.1　PLC 编程环境

PLC 的编程环境是由 PLC 生产厂家设计的，它包含用户环境和能把用户环境与 PLC 系统联结起来的编程软件。只有熟悉了编程环境，了解了编程环境，才能适应编程环境，才能在编程环境中编写出 PLC 的用户程序。

用户环境包括用户数据结构、用户元件区、用户程序存储区、用户参数和文件存储区等。

（1）用户数据结构

用户数据结构分为位数据、字节数据、字数据和混合型数据 4 类。

第一类为位数据，这是一类逻辑量（1 位二进制数），其值为 0 或 1，它表示触点的通或断。触点接通状态为 ON，触点断开状态为 OFF。例如，I0.0 的值表示在输入映像区中的一位二进制数的状态，Q0.0 的值则表示在输出映像区中的一位二进制数的状态。

第二类为字节数据，其位长为 8 位，其数制形式有多种形式。一个字节可以表示 8 位二进制数、两位十六进制数。例如，IB0 的值表示在输入映像区中的连续 8 位二进制数的状态，QB0 的值则表示在输出映像区中的连续 8 位二进制数的状态。

第三类为字数据，其数制、位长和形式都有很多。一个字可以表示 16 位二进制数、4 位十六进制数、4 位十进制数。十进制数据通常都用 BCD 码表示，书写时有时在前面加上 K 字符，例如 K789；十六进制数据，书写时会在前面加上 H 字符，例如 H78F。二进制数，书写时会在前面加上 B 字符，例如 B0111_1000_1111。实际处理时还可选用八进制和 ASCII 码的形式。再如 IW0 表示在输入映像区中的连续 16 位二进制数的状态，QW0 则表示在输出映像区中的连续 16 位二进制数的状态。由于对控制精度的要求越来越高，不少 PLC 开始采用了浮点数，这样极大地提高了数据运算的精度。

第四类为混合型数据，即同一个元件既有位数据又有字数据，例如 T（定时器）和 C（计数器），它们的触点只有 ON 和 OFF 两种状态，是位数据，而它们的设定值和当前值寄

存器又为字数据。

（2）用户数据存储区

用户使用的每个输入/输出端，以及内部的每一个存储单元都称为元件。各种元件都有其固定的存储区（例如输入/输出映像区），即存储地址。给 PLC 中的输入/输出元件赋予地址的过程叫作编址。不同的 PLC 输入/输出的编址方法不完全相同，如 CQM1 的输入端地址可以为 000，001，……通道，输出端地址为 100，101，……通道。

PLC 的内部资源，如内部继电器、定时器、计数器和数据区等，各个不同的 PLC 之间也有一些差异。这些内部资源都按一定的数据结构存储在用户数据存储区，正确使用用户数据存储区的资源才能编好用户程序。

（3）用户程序结构

用户程序是 PLC 的使用者编制的针对具体工程的应用程序。用户程序是线性地存储在 PLC 的存储区间内，它的最大容量也是由具体的 PLC 限制的。

用户程序结构大致可以分为 3 种，一是线性程序，这种结构是把一个工程分成多个小的程序块，这些程序块被依次排放在一个主程序中；二是分块程序，这种结构是把一个工程中的各个程序块独立于主程序之外，工作时要由主程序一个个有序地去调用；三是结构化程序，这种结构是把一个工程中的具有相同功能的程序写成通用功能程序块，工程中的各个程序块都可以随时调用这些通用功能程序块。

2.3.2 PLC 编程语言及编程软件

可编程控制器是通过程序来实现控制的，编写程序时所用的语言就是 PLC 的编程语言，PLC 编程语言有多种，它是用 PLC 的编程语言或某种 PLC 指令的助记符编制而成的。各个元件的助记符随 PLC 型号的不同而略有不同。国际电工委员会（IEC）1994 年 5 月公布 IEC1131－3 标准（PLC 的编程语言标准，也是至今唯一的工业控制系统的编程语言标准）中详细地说明了句法、语义和下述 5 种编程语言：语句表（Statement List，STL）、梯形图（Ladder Diagram，LAD）、功能块图（Function Block Diagram，FBD）、结构文本（Structured Text，ST）、顺序功能图（Sequential Function Chart，SFC）。其中梯形图和语句表编程语言在实际中用的最多，下面着重介绍这两种语言。

（1）梯形图（LAD）

梯形图（LAD）是最常用的 PLC 编程语言。梯形图与继电器的电路图很相似，它是从继电器控制系统原理图演变而来的，是一种类似于继电器控制线路图的一种语言。其画法是从左母线开始，经过触点和线圈，终止于右母线，具有直观、易学、易懂的优点，而且很容易被熟悉继电器控制的工厂电气技术人员所掌握。西门子 PLC 的梯形图具有以下几个特点。

① 梯形图是一种图形语言，沿用继电器控制中的触点、线圈、串并联等专业术语和图形符号。

② 梯形图中的触点有常开触点和常闭触点两种，触点可以是 PLC 输入点接的开关，也可以是内部继电器的触点或内部寄存器、计数器的状态。

③ 触点可以串联或并联，但线圈只能并联，不能串联。

④ 触点和线圈等组成的独立电路称为网络（Network）或程序段。

⑤ 在程序段号的右边可以加上程序段的标题，在程序段号的下边可以加上注释。

⑥ 内部继电器、计数器、寄存器都不能直接控制外部负载，只能作为中间结果供 CPU 内部使用。图 2-4 是启保停电路的梯形图。

（2）语句表（STL）

语句表（STL）类似于计算机的汇编语言，但比汇编语言通俗易懂，是 PLC 的基本编程语言。它用助记符来表示各种指令的功能，指令语句是 PLC 程序的基本元素，多条语句组合起来就构成了语句表。在编程器的键盘上或利用编程软件的语句表格式都可以进行语句表编程。一般情况下语句表和梯形图是可以相互转换的，例如西门子 S7 -300PLC 的 STEP7 编程软件在视图选项中就可以进行语句表和梯形图的相互转换。或者用【Ctrl + 1/2】组合键就可以实现语句表和梯形图的相互转换。要说明的是部分语句表是没有梯形图与之相对应的。启保停电路的梯形图所对应的语句表如图 2-5 所示。

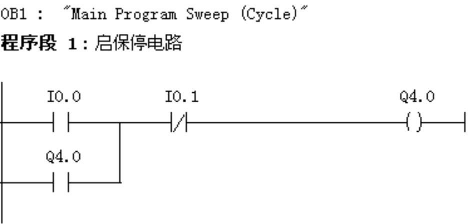


图 2-4　启保停电路梯形图

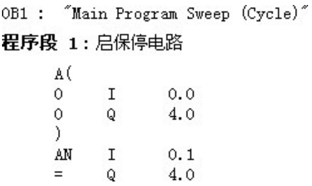


图 2-5　语句表

（3）编程软件

编程器是 PLC 重要的编程设备，它不仅可以用于编写程序，还可以用于输入数据，以及检查和监控 PLC 的运行。在一般情况下，编程器只在 PLC 编程和检查时使用，在 PLC 正式运行后往往把编程器卸掉。

随着计算机技术的发展，PLC 生产厂家越来越倾向于设计一些满足某些 PLC 的编程、监控和设计要求的编程软件，这类编程软件可以在专用的编程器上运行，也可以在普通的个人计算机上运行。这类编程软件利用了计算机的屏幕大、输入/输出信息量多的优势，使 PLC 的编程环境更加完美。在很多情况下，装有编程软件的计算机在 PLC 正式运行后还可以挂在系统上，作为 PLC 的监控设备使用。比如有下列编程软件。

① OMRON 公司设计的 CX - P 编程软件可以为 OMRON C 系列 PLC 提供很好的编程环境。

② 松下电工设计的 FPWin_GR 编程软件可以为 FP 系列 PLC 提供很好的编程环境和仿真。

③ 西门子公司设计的 STEP 7 Micro/WIN 32 编程软件可以为 S7 -200 系列 PLC 提供编程环境。

④ 西门子公司设计的 SIMATIC Manager 编程软件可以为 S7 -300/400 系列 PLC 提供编程环境。

编程软件在使用前一定要把其装入满足条件的计算机中，同时要用专用的通信电缆把计算机和 PLC 连接好，在确认通信无误的情况下才能运行编程软件。

在编程环境中，可以打开编程窗口、监控程序运行窗口、保存程序窗口和设定系统数据窗口，并进行相应的操作。

（4）仿真软件

随着计算机技术的发展，PLC 的编程环境越来越完善。很多 PLC 生产厂家不仅设计了方便的编程软件，而且设计了相应的仿真软件。只要把仿真软件嵌入到编程软件当中，就可以在没有具体的 PLC 的情况下利用仿真软件直接运行和修改 PLC 程序，使 PLC 的学习、设计和调试更方便、快捷。如西门子公司设计的 S7_PLCSIM 仿真软件就是专门为 S7－300/400 PLC 设计的仿真软件，S7_200SIM 是专门为 S7－200 PLC 设计的仿真软件，利用这些仿真软件可以直接运行 S7－200 和 S7－300/400 的 PLC 程序。

2.3.3 PLC 的工作原理

PLC 是一种工业控制用的计算机，它的外形不像个人计算机，工作方式也与计算机差别很大。编程语言甚至工作原理都与个人计算机都有所不同。

PLC 上电后首先要对硬件和软件进行初始化，当其进入运行状态后，PLC 则采用循环扫描的方式工作。在 PLC 执行用户程序时，CPU 对程序采取自上而下，自左向右的顺序逐次进行扫描，即程序的执行是按语句排列的先后顺序进行的。每一次循环扫描所经历的时间称为一个扫描周期。每个扫描周期又主要包括输入刷新、用户程序执行、输出刷新三个阶段。当 PLC 初始化后，就会重复执行以上三个阶段。在进行用户程序执行阶段时，还包括系统自诊断、通信处理、中断处理、立即 I/O 处理等过程。图 2-6 所示为 PLC 的循环扫描工作过程图。

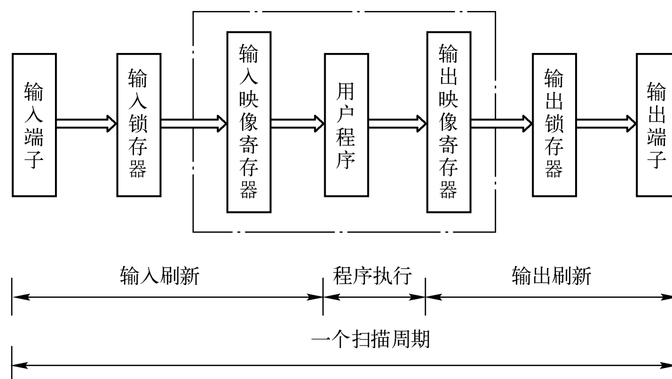


图 2-6　PLC 的扫描工作过程图

（1）输入刷新（采样）阶段

在输入刷新阶段，PLC 以扫描的方式顺序读入所有输入端子的状态，并将此状态存入输入锁存器。如果输入端子上外接电器的触点闭合，锁存器中与端子编号相同的那一位就置“1”，否则为“0”。把输入各端子的状态全部扫描完毕后，PLC 将输入锁存器的内容输入到输入映像寄存器中。输入映像寄存器中的内容则直接反映了各输入端子此刻的状态。这一过程就是输入刷新阶段。随着输入数据输入到输入映像寄存器，标志着输入刷新阶段的结束。所以输入映像寄存器中的内容只是本次输入刷新时各端子的状态。在输入刷新阶段结束后，PLC 接着进入执行用户程序阶段。在用户程序执行和输出刷新期间，输入端子与输入锁存器之间的联系被中断，在下一个扫描周期的输入刷新阶段到来之前，无论输入端子的状态如何变化，输入锁存器的内容都始终保持不变。

（2）用户程序执行阶段

输入刷新阶段结束后，PLC 进入用户程序执行阶段。在用户程序执行阶段，PLC 总是按照自上而下，自左向右的顺序依次执行用户程序的每条指令。从输入映像寄存器中读取输入端子和内部元件寄存器的状态，按照控制程序的要求进行逻辑运算和算术运算，并将运算的结果写入输出映像寄存器中，如果此时程序运行过程中需要读入某输出状态或中间状态，则会从输出映像寄存器中读入，然后进行逻辑运算，运算后的结果再存入输出映像寄存器中。对于每个元件，反映各输出元件状态的输出映像寄存器中所存储的内容，会随着程序的执行而发生变化，当所有程序都执行完毕后，输出映像寄存器中的内容也就固定了下来。

（3）输出刷新阶段

当用户程序的所有指令都执行完后，PLC 就进入输出刷新阶段。输出刷新阶段将输出映像寄存器中的内容存入输出锁存器后，再驱动外部设备工作。与输入刷新阶段一样，PLC 对所有外部信号的输出是统一进行的。在用户程序执行阶段，如果输出映像寄存器的内容发生改变将不会影响外部设备的工作，直到输出刷新阶段将输出映像寄存器的内容集中送出，外部设备的状态才会发生相应的改变。

由 PLC 的工作过程可以看出，在输入刷新期间如果输入变量的状态发生变化，则在本次扫描过程中，改变的状态会被扫描到输入映像寄存器中，在 PLC 的输出端也会发生相应的变化。如果变量的状态变化不是发生在输入刷新阶段，则在本次扫描期间 PLC 的输出保持不变，等到下一次扫描后输出才会发生变化。也就是说只有在输入刷新阶段，输入信号才被采集到输入映像寄存器中，其他时刻输入信号的变化不会影响输入映像寄存器中的内容。

由于 PLC 采用循环扫描的工作方式，并且对输入、输出信号只在每个扫描周期的 I/O 刷新阶段集中输入和集中输出，所以必然会产生输出信号相对输入信号的滞后现象。扫描周期越长，滞后现象就越严重。但是一般扫描周期只有十几毫秒，因此在慢速控制系统中，可以认为输入信号一旦发生变化就能立即进入输入映像寄存器中，其对应的输出信号也可以认为是会及时发生变化的。当某些设备需要输出对输入做出快速响应时，可以采取快速响应模块、高速计数模块以及中断处理等措施来尽量减少滞后时间。

2.4 PLC 产品概况及发展趋势

美国是 PLC 生产大国，在美国注册的 PLC 厂商已超过百家。其中著名的厂家有罗克韦尔公司、通用电气（GE）公司、莫迪康（MODICON）公司、德州仪器（TI）公司、西屋电气公司等。其中罗克韦尔公司是美国最大的 PLC 制造商，其产品约占美国 PLC 市场份额的 50% 左右。

1973 ~ 1974 年，德国和法国也都相继研制出自己的可编程控制器，德国西门子公司（SIEMENS）于 1973 年研制出欧洲第一台 PLC，型号为 SIMATIC S4。德国的西门子（SIEMENS）公司、AEG 公司以及法国的施耐德（TE）公司、瑞士的 SELECTRON 公司等是欧洲著名的 PLC 制造商。西门子公司的产品以其优良的性能而久负盛名。其 S7 系列的主要产品有 S7 – 200（小型机）、S7 – 300（中型机）、S7 – 400（大型机）等。

1971 年，日本从美国引进了这项新技术，由日历公司研制出日本第一台可编程控制器 DSC – 8。日本的 PLC 生产厂家有 40 余家。其中以小型机最具代表性，如欧姆龙、三菱、松

下、富士、日立、东芝等。在世界小型 PLC 市场中，日本的产品约占有 70% 的份额。在中国，欧姆龙（OMRON）的产品销量居于首位。

我国在 1974 年开始引进 PLC，1977 年应用于工业，但仅仅是初步认识与消化阶段。我国 PLC 的发展过程大致可分为 4 个阶段：20 世纪 70 年代初步认识，80 年代引进试用，90 年代后推广应用。2000 年以后 PLC 生产有一定的发展，小型 PLC 已批量生产，中型 PLC 已有产品出现，大型 PLC 已开始研制。国内产品在价格上占有明显的优势，而在质量上还稍欠缺或不足。主要生产单位有北京和利时系统工程股份有限公司、深圳德维森公司、天津市自动化仪表厂等。目前，国产 PLC 形成产品化的生产企业约 30 多家，国内产品市场占有率不超过 10% 。在 90% 的国内 PLC 市场由国外 PLC 产品占领的今天，国产 PLC 能够脱颖而出，并具有和国外同类产品进行竞争的能力，相信在不久的将来，国产 PLC 将占市场更大份额。

目前 PLC 已经广泛应用到石油、化工、机械、钢铁、交通、电力、采矿、环保等各个领域中，还包括从单机自动化到工厂自动化，从机器人、柔性制造系统到工业控制网络等等。从功能上看，PLC 的应用范围大致包括以下几个方面。

（1）开关量的逻辑控制

开关量逻辑控制是 PLC 最基本、最广泛的应用领域，它取代了传统的继电器电路实现逻辑控制。既可以用于单机控制，也可以用于多机控制及自动化生产线检测。如机床、装配生产线、电镀流水线、运输与检测等方面。

（2）运动控制

通过利用 PLC 的单轴或多轴等位置控制模块、高速计数模块等来控制步进电机或伺服电机，使运动部件以适当的速度来实现平滑的直线运动或圆弧运动。可以用于精密的金属切削机床、装配机械、成型机械、机器人等设备的控制。

（3）模拟量处理和 PID 控制

利用 A－D、D－A 转换模块和智能 PID 模块，实现对生产过程中的温度、压力、液位、流量等连续变化的模拟量进行闭环调节控制。

（4）数据处理

PLC 具有数据处理能力，可以完成算术运算、逻辑运算、数据比较、数据传送、数制转换、数据移位、数据显示和打印、数据通信等功能，还可以完成数据采集、分析和处理任务。数据处理一般应用于大型控制系统，如无人控制的柔性制造系统等。

（5）通信联网

PLC 具有通信功能，既可以对远程 I/O 进行控制，又能实现 PLC 与 PLC、PLC 与计算机之间的通信。PLC 与其他智能设备一起可以构成“集中管理，分散控制”的分布式控制系统，以满足计算机集成制造系统及智能化工厂发展的需要。

PLC 自问世以来经过近 40 年的发展，已经成为很多国家的重要产业。另外在国际市场中，PLC 已经成为最受欢迎的工业控制产品。随着科学技术的发展以及市场需求量的增加，PLC 的结构和功能也在不断地改进。生产厂家不停地将功能更强的 PLC 推入市场，平均 3 到 5 年就更新一次。PLC 的发展趋势主要有以下几个方面。

① 向高速度、大容量方向发展

为了提高 PLC 的处理能力，则要求 PLC 具有更好的响应速度和更大的存储容量。目前，有的 PLC 的扫描速度可以达到 0.1 ms/千步左右。在存储容量方面，有的 PLC 最多可以达到

几十兆字节。

② 向超大型和超小型方向发展

当今中小型 PLC 比较多，为了适应市场的需求，PLC 今后会向着多方向发展，特别是超大型机和超小型机两个方向。现在已经有 I/O 点数达到 14336 点的超大型 PLC，它使用 32 位微处理器，多 CPU 并行工作。小型机由整体式结构向小型模块化结构发展，使之配置更加灵活。为了适应市场的需求，现在已经开发出了超小型 PLC，其最小配置的 I/O 点数为 8 ~ 16 点，以适应单机及小型自动控制的需求。

③ 大力开发智能模块，加强联网通信能力

为了满足各种控制系统的要求，近年来不断开发出许多功能模块，如高速计数模块、温度控制模块、远程 I/O 模块、通信和人机接口模块等。这些智能模块既扩展了 PLC 的功能，又扩大了 PLC 的使用范围。加强 PLC 的联网通信能力是 PLC 技术进步的潮流。PLC 的联网通信分为两类：一类是 PLC 之间的联网通信，另一类是 PLC 与计算机之间的联网通信。

④ 加强故障检测与处理能力

在 PLC 的控制系统故障中，由于 CPU、I/O 接口导致的故障约占 20%，它可以通过 PLC 本身的软硬件来检测和处理。由输入输出设备和线路等外部设备导致的故障约占 80%，所以 PLC 的厂家都致力于研制用于检测外部故障的专用智能模块，进一步提高系统的可靠性。

⑤ 编程语言的多样化

在 PLC 系统结构不断发展的同时，PLC 的编程语言也越来越丰富，功能也不断提高。除了常用的梯形图、语句表语言之外，又出现了面向顺序控制的步进编程语言、面向过程控制的流程图语言、与计算机兼容的高级语言（C 语言、BASIC 语言）等。多种编程语言的并存、互补与发展是 PLC 进步的一种趋势。

2.5 PLC 在工业自动化中的地位和作用

工业自动化系统通常分成三类：一类是控制开关量的逻辑装置，一类是控制慢连续量的过程控制系统，一类是控制快连续量的运动控制系统。在传统上对于这三种控制系统用不同的控制装置。逻辑控制系统通常使用电控装置（电气控制装置即继电器接触器控制柜），过程控制系统通常使用电仪装置（电动单元组合仪表），运动控制系统通常使用电传装置（电气传动控制装置）。所谓三电就是指的是电控、电仪、电传。

PLC 集三电于一体，是一种同时具备逻辑控制功能、过程控制功能、运动控制功能、数据处理功能和联网通信功能的多功能控制器。因此 PLC 及其网络被公认为是现代工业自动化三大支柱（PLC、机器人、CAD/CAM）之一。

2.6 SIMATIC 自动化控制系统

2.6.1 SIMATIC 的组成

SIMATIC（Siemens Automatic）即西门子自动化，它是由一系列的部件组合而成的，包括 SIMATIC PLC、PROFIBUS－DP 分布式 I/O、PROFINET I/O 系统中的分布式 I/O、SIMAT-

IC HMI、SIMATIC NET 和标准工具 STEP7 等，其中 PLC 是它的核心部件。

SIMATIC PLC 包括 S7 系列、M7 系列、C7 系列和 WinAC。S7 系列是传统意义的 PLC 产品，它包括微型 PLC（如 S7－200），小规模性能要求的 PLC（如 S7－300），中、高性能要求的 PLC（如 S7－400）等。

S7－200 PLC 是超小型化的 PLC，是西门子专门应用于小型自动化设备的控制装置，主要包括 CPU22X 系列。它适用于各行各业，各种场合中的自动检测、监测及控制等。S7－200 PLC 的强大功能使其无论单机运行，或连成网络都能实现复杂的控制功能。S7－300 是模块化小型 PLC 系统，能满足中等性能要求的应用。各种单独的模块之间可进行广泛组合构成不同要求的系统。图 2-7 为某款 S7－200 PLC 的外观。

与 S7－200 PLC 相比，S7－300 PLC 是模块化的中小型 PLC 系统，能满足中等性能要求的应用，具备高速的指令运算速度；用浮点数运算，有效地实现了更为复杂的算术运算；带有用户接口的软件工具，方便用户给所有模块进行参数赋值；方便的人机界面服务已经集成在 S7－300 PLC 操作系统内，使人机对话的编程要求大大减少。图 2-8 为西门子 S7－300 PLC 的 CPU 外观。



图 2-7　S7－200 PLC 产品图片



图 2-8　S7－300 PLC 产品图片

S7－400 PLC 是中、高档性能的可编程序控制器。它采用模块化无风扇的设计，可靠耐用，易于扩展，同时可以选用多种级别的 CPU，并配有多种功能模板，这使用户能根据需要组合成不同的专用系统。当控制系统规模扩大或升级时，只要适当地增加一些模板，便能使系统升级以充分满足需要。适合于对可靠性要求极高的大型复杂的控制系统。

SIMATIC S7－1200 PLC 和 SIMATIC S7－1500 PLC 控制器是 SIMATIC PLC 产品家族的旗舰产品。SIMATIC S7－1200 PLC 是西门子公司最新推出的面向离散自动化系统和独立自动化系统的紧凑型自动化产品，定位在原有的 SIMATIC S7－200 PLC 和 S7－300 PLC 产品之间，主要应用在简单控制和单机应用，而 SIMATIC S7－1500 PLC 产品家族为中高端工厂自动化控制任务量身定制，适合较复杂的应用。SIMATIC S7－1200 PLC 是西门子低端 PLC 产品的一记重拳，西门子已经停止除在中国的 S7－200CN 系列以外的 S7－200 生产线，S7－200CN 以其低廉的价格还要争夺发展中国家的自动化市场份额。而在欧美低端市场将全部被 S7－1200 PLC 产品覆盖。在中国有很多厂商相继推出兼容 S7－200 PLC 的模块这也使得西门子在低端市场的份额占去一部分，所以为了降低成本而保住市场还要延续 200CN 系列的

辉煌，西门子将会把最新的通信和控制技术应用在 S7 - 1200 PLC 这款产品上，同样西门子也将会用 S7 - 1200 PLC 这款产品强力打造全球 PLC 中低端市场。图 2-9 和图 2-10 分别为西门子 S7 - 1200 PLC 和 SIMATIC S7 - 1500 的外观。



图 2-9　S7 - 1200 PLC 产品图片



图 2-10　S7 - 1500 PLC 产品图片

M7 系列 PLC 中的 M7 - 300/400 PLC 采用与 S7 - 300/400 PLC 相同的结构，具有 AT 兼容计算机的功能。可以用 C、C + + 等高级语言编程，适用于处理数据量大、对数据管理、显示和实时性有较高要求的系统中。SIMATIC C7 由紧凑型 CPU S7 - 31xC、OP、I/O、通信和过程监控系统组成。它的结构紧凑，面向用户的组态/编程、数据管理与通信集成在一起，具有很高的性价比。WinAC 是在 PC 上实现 PLC 功能的，它突破了传统 PLC 开放性差、硬件昂贵、开发周期长等束缚，可以实现控制、数据处理、通信、人机界面等功能。

DP 是现场总线 PROFIBUS - DP 的简称，ET200 分布式 I/O 可以安装在远离 PLC 的地方，通过 PROFIBUS - DP 总线系统实现 PLC 与分布式 I/O 之间的通信。分布式 I/O 可以减少大量的 I/O 接线。集成了 DP 接口的 CPU 可以用作 DP 网络中的主站。

PROFINET I/O 系统由 I/O 控制器和 I/O 设备组成，它们通过工业以太网互联。集成有 PROFINET 接口的 CPU 和通信处理器可以用作 PROFINET I/O 控制器，I/O 控制器与它的 I/O 设备之间进行循环数据交换。IE/PB 链接器用于将工业以太网和 PROFIBUS 子网连接在一起。I/O 控制器可以通过链接器来访问 DP 从站。

HMI 是人机界面。HMI 的硬件配合 PLC 使用，为用户提供数据、图形和事件显示。主要有文本操作面板 TD200、OP3、OP7、OP17；图形/文本操作面板 OP27、OP37；触摸屏操作面板 TP7、TP27/37、TP170A/B；SIMATIC 面板型 PC670。个人计算机（PC）也可以作为 HMI 硬件使用。HMI 硬件需要经过软件组态才能配合 PLC 使用。

SIMATIC NET 将控制系统中所有的站点连接在一起，可以确保站点之间的可靠通信。

符合通信标准的非 SIMATIC 设备也可以集成到 SIMATIC NET 中。

SIMATIC 的标准工具是 STEP 7，它用于对所有的 SIMATIC 部件进行硬件组态和通信连接组态、参数设置和编程。STEP 7 还有测试、启动、维护、文件建档、运行和诊断等功能。

2.6.2 全集成自动化

“全集成自动化（Totally Integrated Automation，TIA）”是西门子公司于 1997 年提出的崭新的革命性的概念，是西门子自动化为响应市场对工业自动化过程控制系统的可靠性、复杂性、功能的完善性、人机界面的友好性、数据分析、管理的快速性、系统安装调试和运行与维护的方便性提出了愈来愈高的需求而提出的概念。它将所有的设备和系统都完整地嵌入到一个自动控制解决方案中，采用共同的组态和编程、共同的数据管理和共同的通信。全集成自动化思想就是用一种系统完成原来由多种系统搭配起来才能完成的所有功能。应用这种解决方案，可以大大简化系统的结构，减少了大量接口部件，应用全集成自动化可以克服上位机和工业控制器之间，连续控制和逻辑控制之间，集中与分散之间的界限。全集成自动化 TIA 是统一的组态和编程、统一的数据库管理和统一的通信，是集统一性和开放性于一身的自动化技术。全集成自动化的特征主要包括统一的数据管理、统一的组态和编程、统一的通信等。

以 STEP 7 为操作平台，所有软件组态都访问同一个数据库。这种统一的数据库管理机制不仅可以减少系统开发的费用，还可以减少出错的概率，提高系统诊断的效率。各软件可以通过全局变量共享一个统一的符号表。在一个项目中只需在一点对变量进行输入和修改。这不仅降低了系统集成的工作量，而且可以避免出现错误。在工程系统中定义的参数可以通过网络向下传输到现场传感器、执行器或驱动器。

在全集成自动化中所有的西门子工业软件都是可以互相配合的，从而实现了高度统一和高度集成。STEP 7 使系统具有统一的组态和编程方式，统一的数据管理和数据通信方式。全集成自动化还采用统一的集成通信技术，使用国际通行的开放的通信标准。TIA 支持基于互联网的全球信息流动，实现了从现场级、控制级到管理级协调一致的通信。

TIA Portal（TIA 博途）是未来西门子全集成自动化系列所有用于工程、编程和调试自动化设备和驱动系统的基础。包括用于 Simatic 控制器的全新工程师站 Simatic Step 7 V11，以及用于 Simatic HMI（人机界面）以及过程可视化应用的 Simatic WinCC V11。

习题

2-1　PLC 的定义及特点是什么？

2-2　简述工业自动化与 PLC 的关系。

2-3　简述 PLC 的分类。

2-4　PLC 主要由哪些部分组成？简述每一部分的作用。

2-5　PLC 的编程语言有几种？

2-6　简述 PLC 的工作原理。

2-7　简述 PLC 的应用以及发展趋势。

2-8　SIMATIC 的组成及各主要作用。

2-9　什么是西门子全集成自动化系统，有哪些优点？

第 3 章　S7 - 300 PLC 硬件系统

前面主要介绍了可编程控制器的基本原理，本章主要介绍 S7 - 300PLC 的硬件组成、电源模块、CPU 模块、接口模块、信号模块、编程设备 PG/PC、人机操作界面、硬件模块的安装、S7 - 300 PLC 的扩展及 I/O 地址分配等内容。通过本章的学习，读者应该掌握 S7 - 300 PLC 的硬件系统知识，为以后的深入学习打下基础。

3.1　S7 - 300 PLC 硬件简介

SIMATIC S 系列 PLC 是德国西门子公司在 S5 系列 PLC 基础上于 1995 年陆续推出的性价比较高的 PLC 系统。它是带有电源、CPU 和 I/O 的一体化单元设备，也可以选择不同的类型的扩展模块，通过选择不同的扩展模块可以将 S7 - 300 PLC 接入 MPI（多点接口）网络、PROFIBUS 网络或以太网中。S7 - 300 PLC 是针对中低性能要求设计的模块化中的小型 PLC，最多可扩展 32 个模块。

S7 - 300 PLC 功能强、速度快、扩展灵活，它具有紧凑的、无槽位限制的模块化结构，其外观构成如图 3-1 所示。

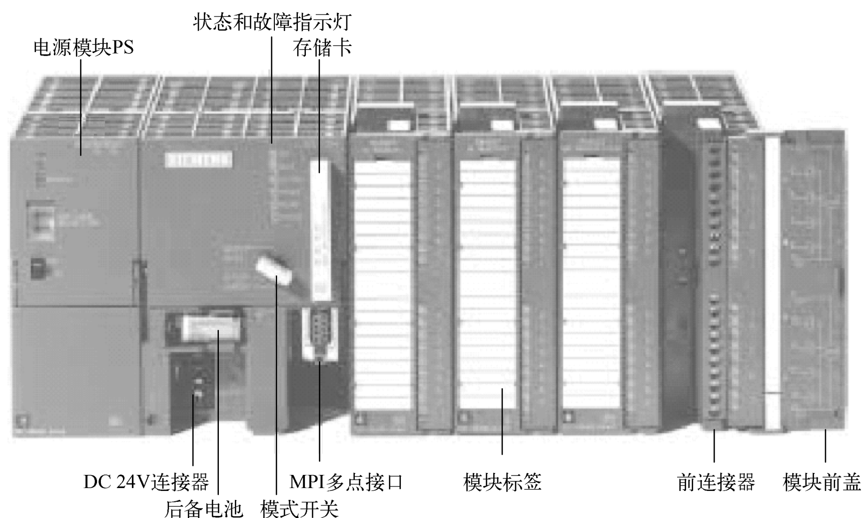


图 3-1　S7 - 300 PLC 的外观结构

3.2　S7 - 300 PLC 机架

机架（Rack）用于安装和连接 PLC 系统的所有模块。

S7 - 300 PLC 的机架是一种 DIN 标准导轨，它是特制不锈钢异型板，其长度有 160，

482，530，830，2000（mm）五种，机架的选型由所使用的模块的总宽度决定，可根据实际需要选择。电源模块、CPU 及其他信号模块都可方便地安装在导轨上，并用螺钉固定。机架中没有背板总线，背板总线集成在模块上，S7－300 PLC 采用背板总线的方式将各模块从物理上和电气上连接起来。除了电源、CPU 和接口模块外，每个机架上最多只能安装 8 个信号模块或功能模块，每个模块只占用一个槽号。如图 3－2 所示为 S7－300 PLC 各模块的连接示意图。

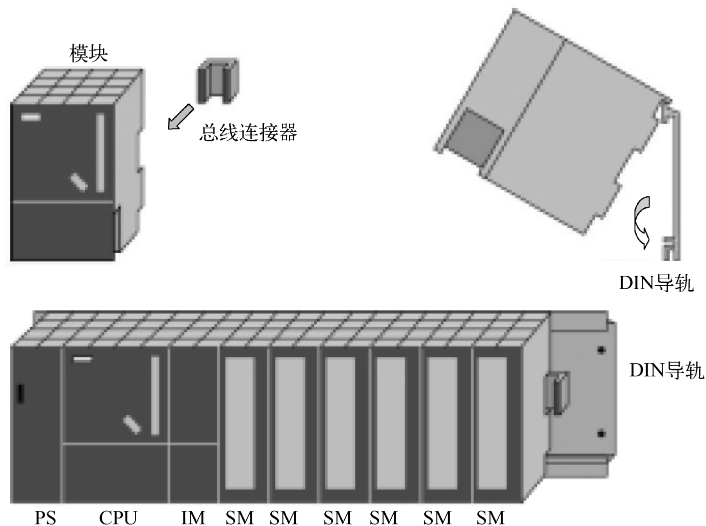


图 3－2　S7－300 PLC 各模块的连接示意图

DIN 导轨用螺钉紧固安装在墙上或机柜中，S7－300 PLC 的所有模块均直接用紧定螺钉紧固在导轨上。即使在有可能发生机械问题的场合，有了 DIN 导轨也可以安全使用 SIMATIC S7－300 PLC 可编程序控制器。安装导轨时，其周围应留有足够的空间，用于散热和安装其他元器件和模块。尤其在系统中有扩展机架时，更应注意每个机架的位置安排。

S7－300 PLC 用背板总线将除电源模块之外的各个模块连接起来。背板总线集成在模块上，除了电源模块，其他模块之间通过 U 型总线连接器相连，后者插在各模块的背后。安装时先将总线连接器插在 CPU 模块上，将后者固定在导轨上，然后依次安装各个模块。

3.3　S7－300 PLC 电源模块

S7－300 PLC 的电源模块（Power Supply，PS）将电源电压转换为 DC 24 V 工作电压，为 CPU 和外围控制电路甚至负载提供可靠的电源。电源模块不仅可以单个供电，也可并联冗余扩充系统容量，进一步提高系统的可靠性。CPU 和扩展接口模块将 24 V 电源转换为 5 V 电源，给背板总线供电，通过背板总线，CPU 监控所有与背板总线连接的接口模块。

电源模块提供了机架和 CPU 内部的供电电源，必须配置于 1 号机架的位置。电源模块用于将 SIMATIC S7－300 PLC 连接到 120/230 V 交流电源，或者连接到直流电源。电源模块将 120 V/230 V 交流电压或者 24/48/72/96/110 V 直流电源转换为 PLC 所需要的 24 V 直流工

作电压。

S7－300 PLC 的电源模块具有 4 种型号：PS305（2A）、PS307（2A）、PS307（5A）和 PS307（10A）。其中 PS305 电源模块为户外型电源模块，输入电压分别为直流 24 V、48 V、72 V、96 V、110 V，输出电压为直流 24 V。

电源模块 PS307 为普通型电源模块，输入电压分别为交流 120 V、230 V，输出电压为直流 24 V，比较适合大多数应用场合。PS307 电源模块的输入和输出之间有可靠的隔离。

电源模块是通过电缆和 CPU 及其他模块之间进行连接供电的，而不是通过背板总线与其他模块进行连接。

S7－300 PLC 中，除了使用 CPU 模块的电源外，其他模块所需的电源是由背板总线提供的，一些模块还须从外部负载电源供电。在组建 S7－300 应用系统时，考虑每个模块的电流耗损量和功率损耗是非常重要的。

一个实际的 S7－300 PLC 系统中，在确定的所有的模块后，要选择合适的电源模块。所选定的电源模块的输出功率必须大于 CPU 模块、所有 I/O 模块、各种智能模块等总消耗的功率之和，并且要留有 30% 左右的裕量。而且，当同一电源模块同时为主机单元和扩展电源供电时，要保证从主机电源到最远一个扩展电源的线路压降不超过 0. 25 V。通常，对于多机架系统，每个机架有一个电源模块。

3. 4　S7－300 PLC CPU 模块

3. 4. 1　CPU 模块的分类

S7－300 系列 PLC 的 CPU 元器件封装在一个牢固而紧凑的塑料机壳内，面板上有状态和故障指示 LED、模式选择开关和通信接口。存储器插槽可以插入多达数兆字节的 Flash EPROM 微存储器卡（简称为 MMC），用于掉电后程序和数据的保存。

S7－300 PLC 有许多种不同型号的 CPU，不同类型的 CPU 具有不同的技术规范和性能参数。每种 CPU 都对应一个型号，比如 CPU 31xC－2DP，其中 31x 表示 CPU 序号，由低到高功能逐渐增强；31x 后面的那位字母表示 CPU 类型，C 表示紧凑型，T 表示技术功能型，F 表示故障安全型；2 代表 CPU 所具有的通信接口个数；最后的 DP 表示通信接口类型，DP 表示 PROFIBUS DP 接口，PN 表示 PROFINET 接口，PtP 表示点对点接口。按性能等级划分，可以涵盖各种应用范围。

S7－300 系列 PLC 的 CPU 按照功能主要有以下几种。

（1）紧凑型 CPU

S7－300 PLC 有 6 种紧凑型 CPU，分别是 CPU312C、CPU313C、CPU313C－PtP、CPU313C－2DP、CPU314C－PtP 和 CPU314－2DP。这些 CPU 的共同特点是带有集成的数字量输入和输出或兼有模拟量的输入和输出，CPU 运行时需要存储卡。多数 CPU 都适用于具备较高要求的系统。型号中带“PtP”的 CPU 除编程端口外还带有第二个串口；型号中带有“2DP”的 CPU 带有 PROFIBUS DP 主站/从站接口。紧凑型 CPU 模块如图 3-3 所示，非紧凑型 CPU 模块如图 3-4 所示。



图 3-3　紧凑型 CPU 模块



图 3-4　非紧凑型 CPU 模块

（2）标准型 CPU

适用于大中规模的 I/O 配置的系统，对二进制和浮点数有较高的处理性能。标准型 CPU 包括 CPU312、CPU313、CPU314、CPU315 -2DP、CPU315 -2PN/DP、CPU317 -2DP、CPU317 -2PN/DP 和 CPU319 -3PN/DP。型号中带有 PN/DP 的 CPU 有一个 PROFINET 接口和一个 MPI/DP 接口。标准型 CPU 模块如图 3-5 所示。



图 3-5　标准型 CPU 模块

（3）技术功能型 CPU

CPU315T -2DP 和 CPU317T -2DP 有极高的处理速度，用于对 PLC 性能以及运动控制功能具有较高要求的设备。除了准确的单轴定位功能以外，还适用于复杂的同步运动控制，一个通信接口是 DP/MPI 接口，另一个是 DP（DRIVE）接口用于连接带 PROFIBUS 接口的驱动系统。技术功能型 CPU 还有本机集成的 4 点数字量输入和 8 点数字量输出，使用标准的编程语言编程，无须专用的运动控制系统语言。

（4）故障安全型 CPU

故障安全型 CPU 包括 CPU315F -2DP、CPU315F -2PN/DP、CPU317F -2DP 和 CPU317F -2PN/DP。它们用于组成故障安全型自动化系统，以满足安全运行的需要，使用内置的 DP 接口和 PROFISAFE 协议，可以在标准数据报文中传输带有安全功能的用户数据。不需要对故障安全 I/O 进行额外的布线，就可以实现与故障安全有关的通信。

（5）SIPLUS 户外型 CPU

SIPLUS 户外型 CPU 包括 SIPLUS 紧凑型 CPU、SIPLUS 标准型 CPU 和 SIPLUUS 故障安全型 CPU。这些模块可以在环境温度 -25℃ ~ +70℃和有害气体的环境中运行。

（6）高端型 CPU

高端型 CPU 包括 CPU317 -2DP、CPU318 -2DP 等，具有大容量程序存储器以及 PROFIBUS - DP 主/从接口，可以用于大规模的 I/O 配置，建立分布式 I/O 结构。

各种不同型号 CPU 的具体性能指标请参考手册《SIMATIC S7 -300 可编程控制器》。

3.4.2　CPU 面板介绍

S7 -300 PLC 的 CPU 种类繁多，具有不同的功能，所以面板也不是完全相同的。如图 3-6 所示分别为不同时期的 CPU314 面板，图 3-6 中左图是 2002 年 10 月之前生产的 CPU314，

图 3-6 中右图是 2002 年 10 月之后生产的 CPU314。

S7 -300 系列 PLC CPU 模块的面板上有状态和故障指示 LED、模式选择开关和通信接口等。大多数 CPU 还有后备电池盒，存储器卡插座可以插入多达数兆字节的 Flash EPROM 微存储卡（简称为 MMC），用于掉电后程序和数据的保存。

（1）卡槽

Flash EPROM 微存储卡用于在断电时保存用户程序和某些数据，它可以扩展 CPU 的存储器容量，也可以将有些 CPU 的操作系统包括在 MMC 中，这对于操作系统的升级是非常方便的。MMC 用作装载存储器或便携式保存媒体，它的读写直接在 CPU 内进行，不需要专用的编程器。由于 CPU 31xC 没有安装集成的装载存储器，在使用 CPU 时必须插入 MMC，因此必须在购买 CPU 的同时也配置 MMC，CPU 与 MMC 是分开订货的。否则 CPU 将无法工作。插入存储卡前，把 CPU 切换到 STOP 状态，或关断电源。MMC 存储卡如图 3-7 所示。



图 3-6 不同时期的 CPU314 面板



图 3-7 MMC 卡外观

（2）状态与故障指示灯 LED

CPU 模块面板上的 LED（发光二极管）的意义见表 3-1。

表 3-1 S7 -300 PLC CPU 的 LED

指示灯	颜色	说明	指示灯	颜色	说明
SF	红色	系统出错/故障显示	FRCE	黄色	有输入/输出处于强制的状态
BF	红色	通信接口的总线故障	RUN	绿色	CPU 处于运行模式
DC 5V	绿色	5V 电源正常	STOP	黄色	CPU 处于停止模式

CPU 处于 RUN 模式时 RUN LED 亮；启动期间以 2 Hz 的频率闪亮；HOLD 状态时以 0.5Hz 的频率闪亮。CPU 处于 STOP、HOLD 状态或重新启动时 STOP LED 常亮；请求存储器复位时以 0.5 Hz 的频率闪动，正在执行存储器复位时以 2 Hz 的频率闪动。

（3）CPU 的操作模式

CPU 有四种操作模式：STOP（停机）、STARTUP（启动）、RUN（运行）和 HOLD（保持）。在所有的模式中，都可以通过 MPI 接口与其他设备通信。

① STOP（停机）模式：可以用模式选择开关在 STOP 位置时，PLC 上电后自动进入 STOP 模式。在该模式下不执行用户程序，可以接收全局数据和检查系统。

② RUN（运行）模式：执行用户程序，刷新输入和输出，处理中断和故障信息服务。

③ HOLD 模式：在启动和 RUN 模式执行程序时遇到调试用的断点，用户程序的执行被挂起（暂停）定时器被冻结。

④ STARTUP（启动）模式：可以用模式选择开关或 STEP7 启动 CPU。如果模式选择开关在 RUN 或 RUN－P 的位置，通电时自动进入启动模式。

老式的 CPU 使用钥匙开关来选择操作模式，它还有一种 RUN－P 模式，允许在运行时读出和修改程序。操作时需要插入钥匙，用于设置 CPU 当前的运行方式。钥匙拨出后，就不能改变操作方式。这样可以防止未经授权的人员非法删除或改写用户程序。仿真软件 PLCSIM 的仿真 CPU 也有 RUN－P 模式，这些监控功能只能在 RUN－P 的模式下进行。

（4）CPU 模式选择开关

① RUN（运行）位置：CPU 执行用户程序。

② STOP（停止）位置：CPU 不执行用户程序。

③ MRES（复位存储器）：可使 CPU 复位。

有以下情况执行 CPU 存储器复位。

a. 在第一次启动前。

b. 在新的完整的用户程序下载前。

c. 如果 CPU 要求存储器复位时（STOP LED 闪烁）。

用模式开关执行 CPU 存储器复位的操作步骤如下。

a. 合上电源开关。

b. 把开关转到 STOP 位置。

c. 把开关转到 MRES 位置，并保持在这个位置直到 STOP 指示灯再次变亮（大约 3 s）。

d. 把钥匙开关转回 STOP 位置，然后再转到 MRES，直到 STOP 指示灯再次亮 1s。

（5）通信接口

所有的 CPU 模块都有一个 MPI 通信接口，有的 CPU 模块还有 DP 接口或点对点接口，型号中带 PN 的 CPU 模块有一个 PROFINET 工业以太网接口。

MPI 接口用于与其他西门子 PLC、PG/PC、OP 通过 MPI 网络通信。

（6）电源接线端子

电源模块上的 L＋和 M 端子分别是 DC24 V 输出电压的正极和负极。用专用的电源连接器或导线分别连接电源模块和 CPU 模块的 L＋和 M 端子。

（7）CPU 模块的集成 I/O

CPU31xC 模块上有集成的 I/O，集成 I/O 的点数见表 3-2。

表 3-2　紧凑型 CPU 部分技术参数

参数 \ CPU	CPU321C	CPU313C	CPU313C－2PtP	CPU313C－2DP	CPU314C－2DP	CPU314C－PtP
用户内存	16 KB	32 KB	32 KB	32 KB	48 KB	48 KB
最大 MMC	4 MB	8 MB	8 MB	8 MB	8 MB	8 MB
自由编址	YES	YES	YES	YES	YES	YES
DI/DO	256/256	992/992	992/992	992/992	992/992	992/992
AI/AO	64/32	246/124	248/124	248/124	248/124	248/124
处理时间/1K 指令	0.1 ms	0.1 ms	0.1 ms	0.1 ms	0.1 ms	0.1 ms

（续）

CPU 参数	CPU321C	CPU313C	CPU313C－2PtP	CPU313C－2DP	CPU314C－2DP	CPU314C－PtP
位存储器	1024	2048	2048	2048	2048	2048
计数器	128	256	256	256	256/256	256
定时器	128	256	256	256	256	256
集成通信连接 MPI/DP/PtP	Y/N/N	Y/N/N	Y/N/Y	Y/Y/N	Y/Y/N	Y/N/Y
集成 DI/DO	10/6	24/16	16/16	16/16	24/16	24/16
集成 AI/AO	0/0	4＋1/2	0/0	0/0	4＋1/2	4＋1/2

3.4.3 CPU 模块存储器

PLC 的操作系统使 PLC 具有基本的智能，能够完成 PLC 设计者规定的各种工作。用户程序由用户设计，它使 PLC 能完成用户要求的特定功能。用户程序存储器的容量以字节（Byte，B）为单位。

（1）PLC 使用的物理存储器

① 随机存取存储器（RAM）

CPU 可以读出 RAM 中的数据，也可以将数据写入 RAM，因此，RAM 又叫作读/写存储器。RAM 具有易失性，即电源中断后，储存的信息会丢失。

RAM 的工作速度快，价格便宜，改写方便。在切断 PLC 的外部电源后，可以用锂电池来保存 RAM 中存储的用户程序和数据。需要更换锂电池时，由 PLC 发出信号，通知用户。

② 只读存储器（ROM）

ROM 的内容只能读出，不能写入。它具有非易失性，即电源消失后，仍能保存存储的内容。ROM 一般用于存储 PLC 的操作系统。

③ 快闪存储器（Flash EPROM）和 EEPROM

快闪存储器简称为 FEPROM，可电擦除可编程的只读存储器简称为 EEPROM。它们具有非易失性，可以用编程装置对它们编程，兼有 ROM 的非易失性和 RAM 的随机存取优点，但是将信息写入它们所需的时间比 RAM 长的多。它们用于存储用户程序和断电时需要保存的重要数据。

（2）微存储卡

基于 FEPROM 的微存储卡简称为 MMC，用于在断电时保存用户程序和某些数据。

MMC 用于作为 S7、C7 和 ET200S 的 CPU 的装载存储器，程序和数据下载后保存在 MMC 内。如果 CPU 未插 MMC，不能下载 STEP7 的程序和数据。应当注意，不能带电插拨 MMC，否则会丢失程序或损坏 MMC。西门子的 PLC 必须使用西门子专用的 MMC，不能使用数码产品使用的通用型 MMC。

如果对 MMC 中的项目加了密码，但是忘记了密码，只能用西门子专用编程器上的读卡槽或用西门子带 USB 接口的读卡器来删除 MMC 上的程序、数据和密码，这样 MMC 就可以作为一个未加密的空卡使用了。

（3）CPU 的存储区

CPU 的存储区由装载存储器、工作存储器和系统存储器组成。工作存储器类似于计算

机的内存条，装载存储器类似于计算机的硬盘。

① 装载存储器

CPU 的装载存储器用于保存不包含符号地址和注释的逻辑块、数据块和系统数据（硬件组态、连接和模块的参数等）。下载程序时，用户程序（逻辑块和数据块）被下载到装载存储器，符号表和注释保存在编程设备中。在 PLC 上电时，CPU 把装载存储器中的可执行部分复制到工作存储器。在 CPU 断电时，需要保存的数据被自动保存在装载存储器中。

S7－300 PLC 将 MMC 用作装载存储器。现在生产的 S7－300 PLC 的 CPU 没有集成的装载存储器，必须插入 MMC 才能下载和运行用户程序。

S7－400 PLC 的 CPU 有集成的装载存储器（带后备电池的 RAM），也可以用 FEPROM 存储卡或 RAM 存储卡来扩展装载存储器。

系统存储区的分布如图 3-8 所示。

② 工作存储器

工作存储器是集成在 CPU 中的高速存取的 RAM 存储器，用于存储 CPU 运行时的用户程序和数据，例如组织块、功能块、功能和数据块。为了保证程序执行的快速性和不过多的占用工作存储器，只有与程序执行有关的块被装入工作存储器。用模式选择开关复位 CPU 的存储器时，RAM 中的程序被清除，FEPROM 中的程序不会被清除。

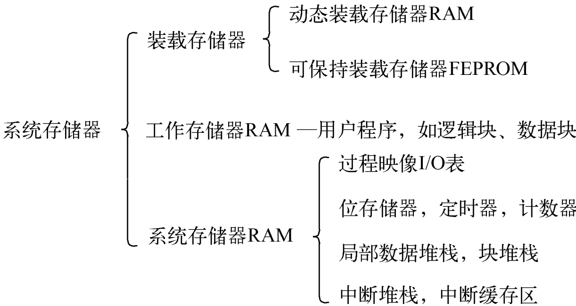


图 3-8　存储区分布

③ 系统存储器

系统存储器是 CPU 为用户程序提供的 RAM 区，用于存储用户程序的操作数据，例如过程映像输入、过程映像输出、位存储器、定时器和计数器、块堆栈、中断堆栈和诊断缓冲区等。系统存储器还包括临时存储器（局部数据堆栈），在程序块被调用时用于存储临时变量。在执行程序块时它的临时变量才有效，执行完后可能被覆盖。

3.5　S7－300 PLC 接口模块

接口模块（Interface Module，IM）用于多机架配置时连接主机架（Central Rack，CR）和扩展机架（Expansion Rack，ER），有 IM360/361/365 三种型号。S7－300 PLC 通过分布式的主机架和 3 个扩展机架，可以操作多达 32 个模块，且运行时无须风扇。

接口模块 IM360 用于 S7－300 PLC 机架 0 的接口，通过连接电缆 368 将数据从 IM360 传送到 IM361，IM360 与 IM361 之间的最大距离为 10 m，IM360 和 IM361 上有指示系统状态和故障的发光二极管，如果 CPU 不确认此机架，则 LED 闪烁，可能是连接电缆没有接好或是串行连接的 IM361 关闭。

接口模块 IM361 采用 24 V DC 电源，用作 S7－300 机架 1 到机架 3 的接口，通过 S7－300 PLC 背板总线的最大电流输出为 0.8A，通过 368 连接电缆将数据从 IM360 传送到 IM361 或从 IM361 传送到 IM360。

接口模块 IM365 是为机架 0 和机架 1 预先组合好的配对模块，总电流为 1.2 A，其中每

个机架最大能使用0.8 A，长1 m 的接线电缆已经固定地连接好。S7 -300 PLC 接口模块的主要特性见表 3-3。

表 3-3　S7 -300 PLC 接口模块的主要特性

应用＼模块	IM360 接口模块	IM361 接口模块	IM365 接口模块
适合于插入 S7 -300 PLC 模块机架	0	1，2，3	0 和 1
数据传输	通过 386 连接电缆，从 IM360 到 IM361	通过 386 连接电缆，从 IM360 到 IM361 或从一个 IM361 到另一个 IM361	通过 386 连接电缆，从一个 IM365 到另一个 IM365
距离	最长 10 m	最长 10 m	1 m，永久链接

3.6　S7 -300 PLC 信号模块

输入/输出模块统称为信号模块（Signal Module，SM），其主要用于信号的输入和输出，使不同的过程信号电压或电流与 PLC 内部的信号电平匹配。对于没有集成 I/O 点或需要扩展 I/O 点的 CPU 如 CPU31x 系列 CPU，则必须用到信号模块进行 I/O 扩展。

按照信号的特性分类，信号模块可分为数字量模块和模拟量模块，主要有数字量（开关量）输入（DI）模块、数字量（开关量）输出（DO）模块、数字量（开关量）输入/输出（DI/DO）模块、模拟量输入（AI）模块、模拟量输出（AO）模块和模拟量输入/输出（AI/AO）模块。

S7 -300 PLC 数字量信号模块包括数字输入模块 SM321、数字输出模块 SM322 和 SM323/ SM327 数字量输入/输出模块等，可用于连接数字传感器和执行元件数字 I/O，使 PLC 灵活地与任务相适应。模拟量信号模块包括模拟输入模块 SM331、模拟输出模块 SM332 和 SM333/ SM337 模拟量输入/输出模块等。通过这些模块可以将模拟传感器和执行元件与 S7 -300 PLC 相连接。此外还有用于调试的 SM374，它是 16 点数字模块，通过旋动面板上的开关，可以自由实现三种输入/输出点数：即输入 16 点、输出 16 点和输入/输出各 8 点。

SIMATIC S7 -300 PLC 的数字输入/输出模块用于连接开关、2 线接近开关、电磁阀、接触器、小功率电动机、灯和电动机启动等，将控制过程的外部数字量电平转化为 S7 -300 的内部信号电平，并将 S7 -300 PLC 内部信号电平转化为控制过程所需的外部信号电平。

模拟 I/O 模块具有下列优点。

① 优化配合。模块可以任意配合所需输入/输出点数量，没有必要增加投资。

② 强大的模拟技术。不同的 I/O 范围和高分辨率允许与众多不同的模拟传感器和执行元件相连。

信号模块结构紧凑，组装简单，接线方便。该模块安装在 DIN 标准导轨上并通过总线连接器与相邻模块相连接。

3.6.1　数字量模块

（1）数字量输入模块 SM321

数字量输入模块用于采集现场过程的数字信号电平，并把它转换为 PLC 内部的信号电

平。一般数字量输入模块连接外部的机械触点和电子数字式传感器。数字量模块的输入输出电缆最大长度为 1000 m（屏蔽电缆）或 600 m（非屏蔽电缆）。

用于采集直流信号的模块称为直流输入模块，其名称含有 VDC，额定输入电压为直流 24 V；用于采集交流信号的模块称为交流输入模块，其名称含有 VAC，额定输入电压为交流 120 V/230 V。如果信号线不是很长，PLC 所处的物理环境较好，电磁干扰较轻，应考虑优先选用以 DC24 V 的直流输入模块。交流输入方式适合于在油雾、粉尘的恶劣环境下使用。

对于用户来说，数字量输入模块 SM321 有四种型号的模块可供选择，分别是 DC16 点输入、DC32 点输入、AC16 点输入和 AC8 点输入模块。

在把数字量输入量模块与电源和 CPU 模块安装在导轨上后，需要连接电缆线为 SM321 供电。

模块上的每个输入点的输入状态是用一个绿色的发光二极管来显示的，输入开关闭合即有输入电压时，二极管点亮。图 3-9 所示为直流 32 点输入对应的端子连接及电气原理图。

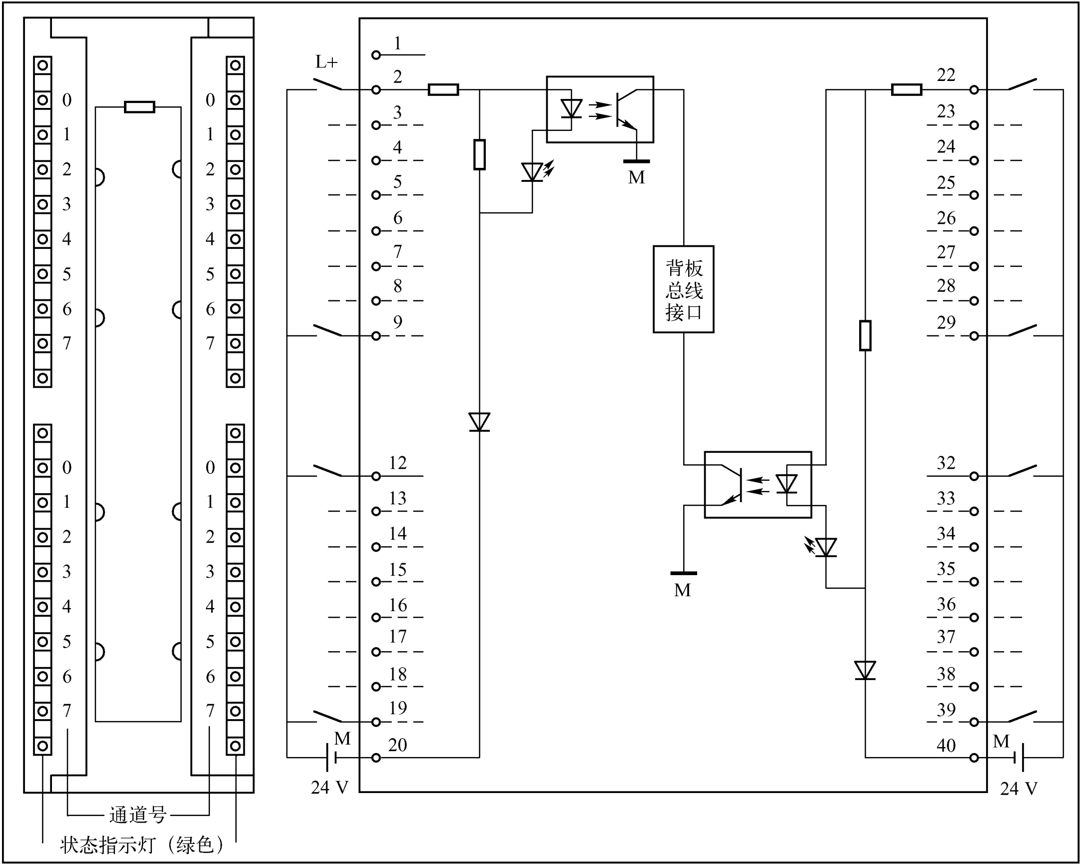


图 3-9　数字量输入模块 SM321 端子连接及电气原理图

（2）数字量输出模块 SM322

数字量输出模块将 PLC 内部信号电平转换成外部过程所需的信号电平，同时具有隔离和功率放大的作用。该模块能连接继电器、电磁阀、接触器、小功率电动机、指示灯和电动机软启动等负载。

按负载回路使用的电源不同，数字量输出模块可以分为直流输出模块、交流输出模块和交直流两用输出模块。按输出开关器件的种类不同，它又可分为晶体管输出方式、晶闸管输出方式和继电器输出方式。

以上两种分类方式又有密不可分的关系。晶体管输出方式的模块只能带直流负载，属于直流输出模块；晶闸管输出方式的模块属于交流输出模块；继电器输出方式的模块属于交直流两用输出模块。从响应的速度上看，晶体管响应最快，继电器响应最慢；从安全隔离效果及应用灵活性角度看，继电器输出型的性能最好。

在一般情况下，用户多采用继电器型的数字输出模块，而它的价格也相对高一些。继电器输出模块的额定负载电压范围较宽，输出直流最小是 DC24 V，最大可到 DC120 V；输出交流的范围是 AC48 ~ 230 V。

数字量输出模块 SM322 有多种型号输出模块可供选择，常用模块有 8 点晶体管输出、16 点晶体管输出、32 点晶体管输出、8 点可控硅输出、16 点可控硅输出、8 点继电器输出和 16 点继电器输出。模块的每个输出点有一个绿色发光二极管显示输出状态，输出逻辑“1”时，二极管点亮。

（3）数字量 I/O 模块 SM323

SM323 模块有两种类型，一种是带有 8 个共地输入端和 8 个共地输出端，另一种是带有 16 个共地输入端和 16 个共地输出端，两种特性相同。图 3-10 是 8 个共地的输入端、输出

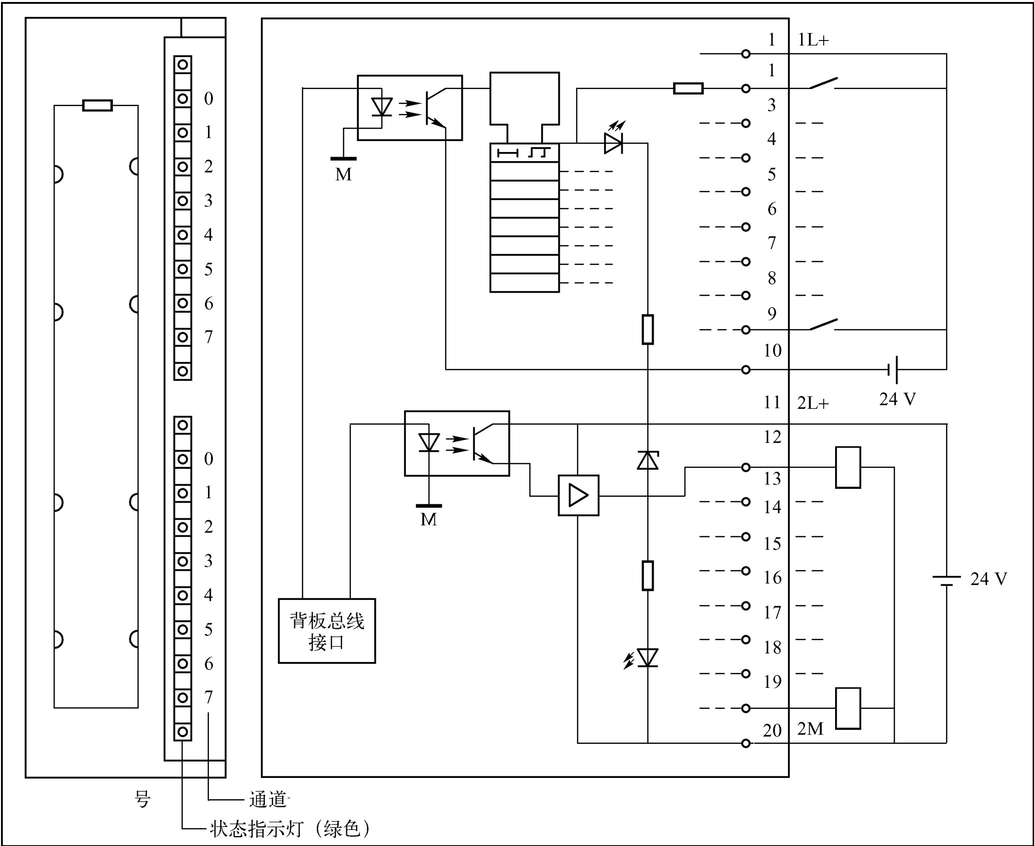


图 3-10　SM323 模块端子连接及电气原理图

端 SM323 模块的端子连接及电气原理图，端子 1～10 用于输入，端子 11～20 用于输出。I/O 额定负载电压 24 V DC，输入电压“1”信号电平为 11～30 V，“0”信号电平为 -3～+5 V，I/O 通过光耦与背板总线隔离。在额定输入电压下，输入延迟为 1.2～4.8 ms。输出具有电子短路保护功能。

3.6.2 模拟量模块

在生产过程中，存在大量的物理量，例如速度、旋转速度、pH 值、黏度、有功功率和无功功率等。有的是非电量，例如温度，压力、流量、液位、物体的成分。为了实现自动控制，这些模拟量信号需要被 PLC 处理。模拟量输入模块 SM331 用于连接电压和电流传感器、热电偶、电阻器和电阻式温度计，将扩展过程中的模拟信号转换为 S7-300 内部处理用的数字信号。模拟量输出模块 SM322 用于将 S7-300 PLC 与执行元件相连，将数字输出值转换为模拟信号。模拟量输入/输出 SM334、SM335 兼有模拟输入和模拟输出功能。

S7-300 PLC 的模拟量 I/O 模块包括模拟量输入模块 SM331、模拟量输出模块 SM322、模拟量输入/输出模块 SM334 和 SM335。

（1）模拟量输入模块 SM331

模拟量输入模块 SM331 目前有三种规格型号，即 8AI×12 位模块、2AI×12 位模块和 8AI×16 位模块。

SM331 主要由 A-D 转换部件、模拟切换开关、补偿电路、恒流源、光电隔离部件、逻辑电路等组成。A-D 转换部件是模块的核心，其转换原理采用积分方法，被测模拟量的精度是所设定的积分时间的正函数，即积分时间越长，被测量的精度越高。SM331 可选四档积分时间：2.5 ms、16.7 ms、20 ms 和 100 ms，相对应表示的精度为 9、12、12 和 14（如图 3-11 所示）。每一种积分时间有一个最佳的噪声抑制频率 f，以上四种积分时间分别对应 400 Hz、60 Hz、50 Hz、10 Hz。例如 A-D 的积分时间设定为 20 ms，则它的转换精度为 12 位，此时对频率为 50 Hz 的噪声干扰有很强的抑制作用。在我国为了抑制工频及其谐波的干扰，一般采用 20 ms 的积分时间。

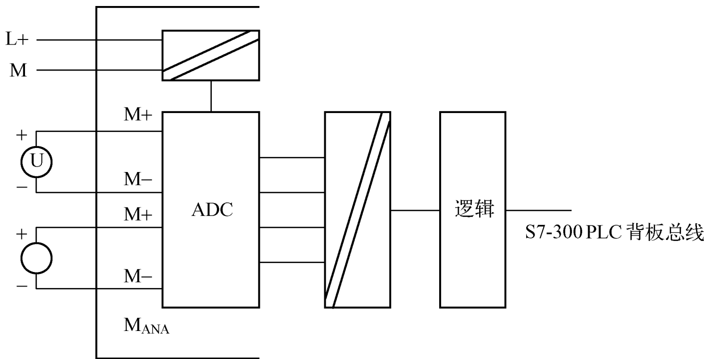


图 3-11　SM331 与电压型传感器的连接

（2）模拟量输出模块 SM332

模拟量输出模块 SM332 目前有三种规格型号，即 4AO×12 位模块、2AO×12 位模块和 4AO×16 位模块，分别为 4 通道的 12 位模拟量输出模块、2 通道的 12 位模拟量输出模块、

4 通道的 16 位模拟量输出模块。

SM332 可以输出电压，也可以输出电流。在输出电压时，可采用 2 线回路和 4 线回路两种方式与负载相连。采用 4 线回路获得比较高的输出精度（如图 3-12 所示）。

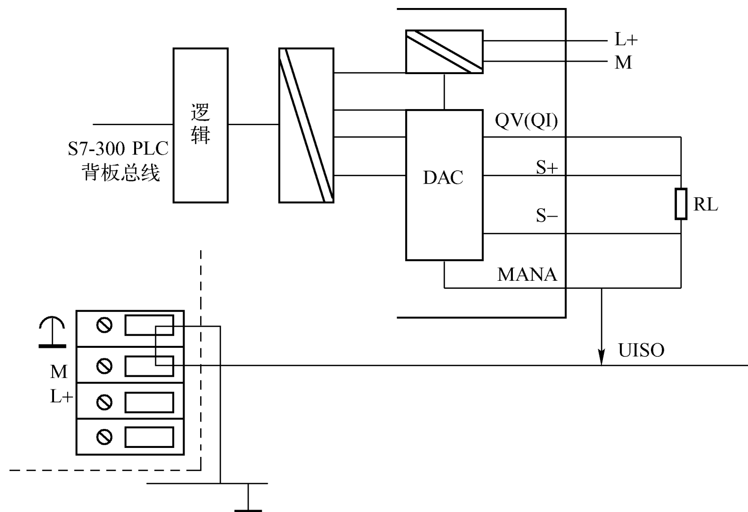


图 3-12　通过 4 线回路将负载与隔离的模出模块相连

（3）模拟量输入模块的量程卡

模拟量输入模块的输入信号类型用量程卡（或称为量程模块）来设置。量程卡安装在模拟量输入模块的两侧，每两个通道为一组，每组共用一个量程卡，图 3-13 中的模块有 8 个通道，因此有 4 个量程卡。

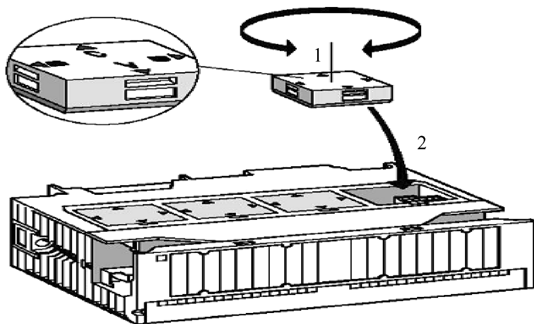


图 3-13　量程卡

（4）模拟量输入模块的输出值计算对应的物理量

① 模拟量转换后的模拟值表示方法

模拟量输入/输出模块中模拟量对应的数字称为模拟值，模拟值用 16 位二进制补码（整数）表示。最高位（第 15 位）为符号位，正数的符号位为 0，负数的符号位为 1。

模拟量经 A-D 转换后得到的数值的位（转换精度）可以设置为 9~16 位（与模块的型号和组态有关），如果小于 16 位（包括符号位）则转换值被自动左移，使其最高位（符号位）在 16 位字的最高位，模拟值左移后未使用的低位则填入“0”，这种处理方法称为“左

对齐”设模拟值的精度为 12 位加符号位，左移 3 位后未使用的低位（第 0 ~ 2 位）为 0，相当于实际的模拟量值乘以 8。这种处理方法使模拟值与模拟的关系与组态的 A/D 转换的位数无关，便于对模拟值的后续处理。

表 3-4 给出了模拟输入模块的模拟值与以百分数表示的模拟量之间的对应关系，其中最重要的关系是双极性模拟量量程的上、下限（100% 和 -100%）分别对应于模拟值 27648 和 -27648。单极性模拟量量程的上、下限（100% 和 0）分别对应于模拟值 27648 和 0。

表 3-4　SM331 模拟输入模块的模拟值

范围	双极性						单极性					
	百分比	十进制	十六进制	±5 V	±10 V	±20 mA	百分比	十进制	十六进制	0 ~ 10 V	0 ~ 20 mA	4 ~ 20 mA
上溢出	118. 515%	32767	7FFFH	5. 926 V	11. 851 V	23. 70 mA	118. 515%	32767	7FFFH	11. 852 V	23. 70 mA	22. 96 mA
超出范围	117. 589%	32511	7EFFH	5. 879 V	11. 759 V	23. 52 mA	117. 589%	32511	7EFFH	11. 795 V	23. 52 mA	22. 81 mA
正常范围	100. 000%	27468	6C00H	5 V	10 V	20 mA	100. 000%	27468	6C00H	10 V	20 mA	20 mA
	0%	0	0H	0 V	0 V	0 mA	0%	0%	0H	01 V	0 mA	4 mA
	-100%	-27648	9400H	-5 V	-10 V	-20 mA	—	—	—	—	—	—
低于范围	-117. 593%	-32512	8100H	-5. 879 V	-11. 759 V	-23. 52 mA	-17. 593%	-4864	ED00H	-	-3. 52 mA	1. 185 mA
下溢出	-118. 519%	-32768	8000H	-5. 926 V	-11. 851 V	-23. 70 m	—	—	—	—	—	—

② 转换举例

一个压力检测系统，压力传感器的量程为 0 ~ 15 MPa，输入信号为 4 ~ 20 mA。模拟量输入模块的量程设置为 4 ~ 20 mA，转换后的模拟值为 0 ~ 27648。设转换后得到的数字为 N，试求以 kPa 为单位的压力值。

解：0 ~ 15 MPa（0 ~ 15000 kPa）对应于转换后的数字 0 ~ 27648，转换公式为：

$$P = 15000 \times N/27648 \text{ kPa}$$

注意：在运算时一定要先乘后除，否则会损失原始数据的精度。

3.6.3　数字量仿真模块

仿真模块 SM374 常用于调试程序和实验中，直接用模块的开关来模拟数字量输入/输出信号，同时用 LED 灯显示输入/输出状态。

仿真模块 SM374 可以仿真 16 点输入、16 点输出、8 点输入和 8 点输出的数字量模块。图 3-14 是 SM374 的前视图，用螺丝刀改变面板中间开关的位置，即可仿真所需的数字量模块。仿真模块没有列入 S7 组态工具的模块目录中，也即 S7 的结构不承认仿真模块的工作方式，但组态时可以填入被仿真模块的代号。例如，组态时若 SM374 仿真 16 点输入的模块，就填入 16 点数字量输入模块的代号：6ES7 311 - 1BH00 - 0AA00；若 SM374 仿真 16 点输出的模块，就填入 16 点数字量输出模块的代号：6ES7 322 - 1BH00 - 0AA00。SM374 面板上有 16 个开关，用于输入状态的设置，还有 16 个绿色 LED，用于指示 I/O 状态。使用 SM374 后，PLC 应用系统的模拟调试变得简单而方便。

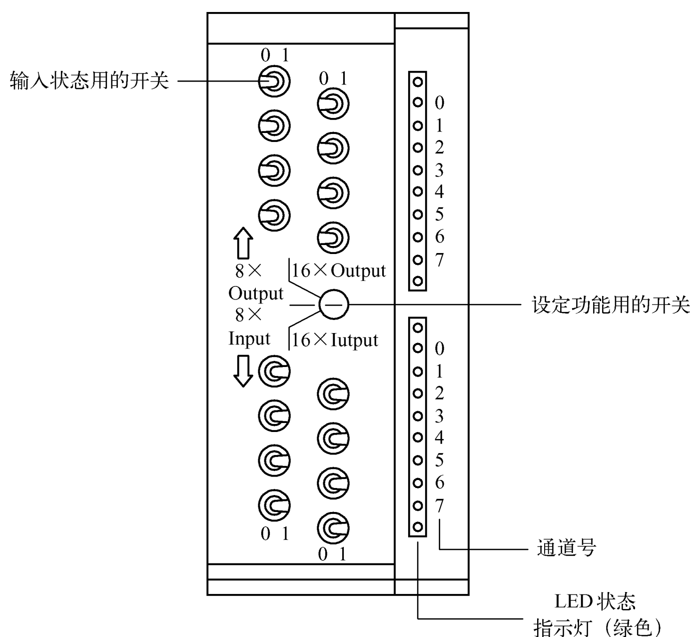


图 3-14　仿真模块 SM374 前视图

3.7　其他模块

（1）通信处理器模块（CP3X）

通信处理模块（Communication Processor，CP）为 PLC 应用系统接入 PROFIBUS 网络、工业以太网和 AS－i 等网络提供了极大的便利。通过集成在 STEP 7 中的参数化工具可进行简便的参数设置。

CP340：用于点对点连接的通信模板。

CP341：用于点对点连接的通信模板。

CP343－1：用于连接工业以太网的通信模板。

CP343－2：用于 AS 接口的通信模板。

CP342－5：用于 PROFIBUS DP 的通信模板。

CP343－5：用于连接 PROFIBUS FMS 的通讯模板。

（2）特殊功能模块（FM）

FM 350－1，FM 350－2：计数器模板。

FM 351：用于快速/慢速驱动的定位模板。

FM 353：用于步进电动机的定位模板。

FM 354：用于伺服电动机的定位模板。

FM 357－2：定位和连续通道控制模板。

SM 338：超声波位置探测模板。

SM 338：SSI 位置探测模板。

FM 352：电子凸轮控制器。

FM 352－5：高速布尔运算处理器。

FM 355：PID 模板。

FM 355－2：温度 PID 控制模板。

（3）Ex 系列输入/输出模块

Ex 系列模块，即防爆型模块，包括 Ex 数字量输入/输出模块和 Ex 模拟量输入/输出模块，主要用于有潜在危险的化工厂场合，将外部的本质——安全设备（用于有爆炸危险区域的传感器和执行器）与 PLC 非本质——安全内部回路隔离。

（4）F 系列输入/输出模块

F 系列输入/输出模块，即故障安全型模块，包括 SM326F 数字量输入——安全集成模块、M326F 数字量输出——安全集成模块、M326F 模拟量输入——安全集成模块，这些模块具有故障安全运行的集成安全功能，在 ET 200M 分布式 I/O 或 S7－300F 中使用。用于连接有爆炸危险区域的设备。

3.8　编程设备 PG/PC

PC/MPI 适配器使用户能在 PC 和 PLC 之间建立数据联系，图 3-15 为其外观。有两种接口：RS－232 接口和 USB 接口。若在 PC 上安装通信卡，能使 PC 和 PLC 之间通过网络进行通信。有连接到 PLC 的 MPI、PROFIBUS、工业以太网的各种通信卡。

根据实际需求，用户可以选择不同的接口。基本的思想是软件设置必须与选取的硬件相配合。这里的软件设置指在“Setting PG/PC Interface”窗口中的设置，通过在控制面板中双击“Setting PG/PC Interface”图标打开；或者可以在 SIMATIC 管理器窗口中，执行菜单命令“Option/Setting PC/PG Interface”打开。只有在安装好 STEP7 软件的 PC 中才会在“控制面板”中出现这个图标。

设置“设置 PG/PC 接口”对话框中的 PC/PG 接口参数，方法如下。

①“应用程序访问点”设置为“S7ONLINE”，如图 3-16 所示。



图 3-15　PC/MPI 适配器外观



图 3-16　“设置 PG/PC 接口”对话框

② 在“所分配的接口参数集”的表中，选择所需接口参数为“PC Adapter（MPI）”或者为 CP5611（MPI）。

③ 可以单击“属性”打开“属性 – PC Adapter（MPI）”对话框，根据用户使用的编程电缆设置正确的 PC 接口。如图 3–17 所示。

若要把编程设备连接到 PLC 系统的 PROFIBUS 网络中，即网络中需挂有多个 PLC，它们有不同的 PROFIBUS 地址，那么下载硬件组态和用户程序时，则需要在上述第②步设置为“CP5611（PROFIBUS）”。如果“所分配的接口参数集”项中没有显示出所需要的接口参数，那么用户必须单击“选择”按钮先安装模板或协议，这样，接口参数就会自动生成。

同样，若要把编程设备连接到 PLC 系统的 MPI 网络中，则在第②步设置为“CPI5611（MPI）”。

若要把编程设备作为上位机的监控（比如安装组态软件 WinCC），那么在上述第①步设置为“MPI（WINCC）”。



图 3–17 “属性 – PC Adapter（MPI）”对话框

3.9 人机操作界面

3.9.1 人机界面的概念

人机界面是操作员与 PLC 之间进行人机对话和相互作用的接口设备。近年来，人机界面设备的价格大幅度下降，其应用越来越广泛，已经成为现代工业控制系统不可缺少的设备之一。

过去用按钮、开关和指示灯等作为人机界面装置，它们提供的信息量少，而且操作困难，需要熟练的操作人员来操作。如七段数字显示器来显示数字，用拨码开关来输入参数，占用的 PLC 的 I/O 点数多，硬件成本高，有时还需要自制印制电路板。

HMI 是操作员通过输入单元写入工作参数或输入操作命令，与控制系统之间进行对话和相互作用的数字设备。人机界面装置是操作人员与 PLC 之间双向沟通的桥梁，很多工业被控对象要求控制系统具有很强的人机界面，用于实现操作人员与计算机控制系统之间的对话和相互作用。人机界面装置用于显示 PLC 的 I/O 状态和各种系统信息，接收操作人员发出的各种命令和设置的参数，并将它们传送到 PLC。人机界面装置一般安装在控制屏上，能够适应恶劣的现场环境，可靠性好。在环境较好的控制室内可以用计算机作为人机界面装置。

3.9.2 人机界面的功能

人机界面最基本的功能是显示现场设备中开关量的状态和寄存器中数字变量的值，用监控界面向 PLC 发布命令，并修改 PLC 寄存器中的参数。另外还包括如下功能。

① 过程可视化。

② 操作人员对过程的控制。

③ 显示报警。

④ 记录设备生产数据记录。

⑤ 脚本。

⑥ 应用于通信频率高的场合。

3.9.3 WinCC 软件

WinCC 是西门子公司于 1996 年推出的基于 Windows 操作系统的强大的人机接口系统，即人和机器之间的界面系统，广泛应用于过程通信和过程可视化。基于 PC 的操作员控制和监视系统正处于快速发展阶段。WinCC 是 Windows Control Center 的简写，可提供 Windows NT、Windows 2000/XP 和 Windows 2003 标准环境下安全控制系统所需的所有功能。

WinCC 除了支持西门子的自动化系统外，还能与 AB 等其他公司的系统连接；通过 OPC 方式，WinCC 可以与更多的第三方控制器进行通信。

作为 SIMATIC 全集成自动化系统的重要组成部分，WinCC 确保与 SIMATIC S5、S7 系列的 PLC 方便连接和高效通信；WinCC 与 STEP7 编程软件的紧密结合缩短了项目的开发周期；此外，WinCC 还有对 SIMATIC PLC 进行系统诊断的选项，给硬件的维护提供了方便。

WinCC 可以与多种自动化设备及控制软件集成，具有丰富的设置项目、可视窗口和菜单选项，使用方式灵活、功能齐全。用户在友好的界面下进行组态、编程和数据管理，可形成所需的操作界面、监控界面、报警界面、实时趋势曲线、历史趋势曲线及报表打印等功能。WinCC 的主要功能如下。

（1）集成用户管理

使用 WinCC 用户管理器，可以分配和控制组态及运行时的访问权限。系统管理员可以随时建立最多 128 个用户组，并为它们分配相应的访问 WinCC 功能的权限。所有操作员工作站都包括在用户管理范围内，例如 Web Navigator 客户机。

（2）图形系统

WinCC 的图形系统可处理运行时在屏幕上显示的所有输入和输出。使用 WinCC 图形设计器可生成用于工厂可视化和操作的图形。不管是简单的操作和监视任务，还是复杂的管理任务，WinCC 都可以生成个性化组态的用户界面，以期实现安全的过程控制和整个生产过程的优化。

（3）操作与监视

WinCC 通过锁定未授权的访问，可保护过程、归档和避免未经授权的操作员进入。WinCC 可以记录各种变量输入，并带有日期、时钟时间、用户名以及新、旧值之间的比较。

WinCC 系统可提供从图形、按钮、柱状图和控件等直到每个用户对象的各种对象，并支出最多 4096 × 4096 像素的图形显示，包括平移、缩放和界面整理等。

（4）消息和归档系统

SIMATIC WinCC 不仅可以获取过程消息和本地事件，而且还能将这些消息和本地事件信息存储在循环归档系统中，在需要时，通过过滤或分类加以利用。消息可以从某个位导出，也可以直接来自自动化系统的一个报警消息帧，或者是超出极限条件的模拟量报警。用户可对消息进行组态，可自由定义消息结构。WinCC 能区分多达 16 类信息，可以显示干扰状态消息以及几个工厂工段的报警、告警、干扰和差错等。

Microsoft SQL 服务器用于归档消息。在消息序列记录中，可根据时间顺序归档消息，可

打印出当前列队中消息的所有状态变化。在消息归档记录中，可有选择性地生成归档消息的特殊视图。

（5）报表和记录系统

WinCC 有一个集成的记录系统，可记录和打印来自 WinCC 或其他应用程序的数据。在打印报表之前，可将数据作为文件来保存，并可在显示器上进行预览。可分别组态布局，实现灵活打印。

用户能在时间或事件驱动的基础上，通过直接输入来启动一个报表的输出。用户也可以给每个打印作业指定不同的打印机。

（6）开发，易于集成

WinCC 还可以接收来自数据库的数据和 CSV 格式的外部数据；为了以表格或图形方式显示从其他应用程序来的数据，用户可以开发自己的数据报表（Report Provider）。

SIMATIC WinCC 组态简单、高效。例如消息向导可以提供有待确认或修改的默认设置；预览窗口可以显示参数选定后的效果，开发人员通过设定参数，能在非常短的时间内完成高性价比的可行解决方案；创建的图形对象可反复地集成到界面。

3.10 S7－300 PLC 模块的安装与扩展

3.10.1 模块配放原则

要组建一个 S7－300 PLC 的硬件系统，至少要用到导轨、电源模板、CPU 模板、信号模板。有的 CPU 将信号模板集成在 CPU 模板上。整个 S7－300 PLC 的硬件系统中，最多可配置 32 块 I/O 模板，分布在四个机架上，模板之间用 U 型连接器连接，可利用 MPI、PROFIBUS 和工业以太网组成网络，使用 STEP7 组态工具可以对硬件进行组态和设置。

① 模块必须无间隙地插入到机架中，否则背板总线将被中断。

② 在 S7－300 PLC 的硬件系统的安装中，有一些必须遵守的约定。在 0 号机架中，插槽 1 分配给电源模板其不占用分配地址；插槽 2 分配给 CPU 模板，其必须紧靠电源模板，也不分配地址；插槽 3 分配给接口模板。用于连接扩展机架，即使不使用接口模板，CPU 也给接口模板分配逻辑地址；4～11 号插槽可自由分配信号模块 SM、功能模块 FM 和通信模块 CP。

③ 在 1～3 号扩展机架中，1、2、3 号槽位是固定的，即插槽 1 插电源模块或为空、插槽 2 为空、插槽 3 插接口模块。即便只有 1 个主机架，3 号槽位不装 IM 接口模块，也不能装其他模块，可安装占位模块补空位并连续背板总线，也方便以后扩展。

3.10.2 模块安装规范

S7－300 PLC 采用紧凑的、无槽位限制的模块结构。一个 S7－300 PLC 系统由多个模块组成，所有模块安装在机架上，根据需要选择合适的模块组建 S7－300 PLC 系统。S7－300 PLC 既可以水平安装，也可以垂直安装。CPU 和电源必须安装在左侧（水平安装）或底部（垂直安装），其允许的环境温度为 0℃～60℃（水平安装）和 0℃～40℃（垂直安装）。

为了提高模块的安装空间和确保模块散热良好，机架在控制柜中的最小安装间距如下。

① 机架左右为 20 mm。

② 机架上下单层组态安装时，上下为 40 mm；多层组态安装时，上下至少为 80 mm。图 3-18 为 S7-300 单机架安装示意图。

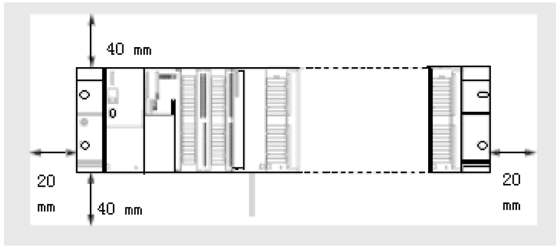


图 3-18　S7-300 单机架安装示意图

3.10.3　模块安装步骤

模块的安装按照以下步骤进行。

（1）安装机架（导轨）

（2）将模块安装在机架上

模块安装时按照电源模块（PS）、CPU 模块、接口模块（IM）、信号模块（SM）、功能模块（FM）、通信模块（CP）的顺序进行。模块背后有总线连接器，安装时先将总线连接器插在模块上，并依次连接后悬挂在导轨上，最后用螺钉进行固定。

（3）电源模块的接线

PS307 电源模块用于将交流电源转换为直流稳压电源，供 CPU 模块和 I/O 模块使用。其额定输出电流分别为 2 A、5 A 和 10 A。

电源模块为系统提供了稳定的 DC24V 工作电源。将市电接入电源模块上的三个端子（L1、N、接地）上，再将通过两个端子（L+和 M）输出的 24 V 电源线引出，为其他模块供电。具体接法如图 3-19 所示。

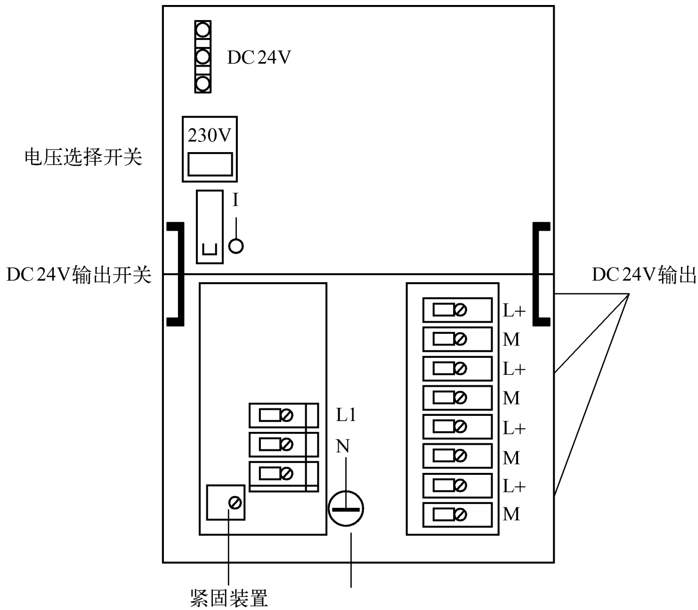


图 3-19　连接电源模块和 CPU

（4）信号模块的接线

前连接器用于将系统中的传感器和执行元件连接至 S7－300 PLC。模块由插入式前连接器与传感器和执行器接线，接好后插入模块。第一次插入连接器时，有一个编码元件与之啮合，该连接器以后就只能插入同样类型的模块。更换模块时，前连接器的接线状况无须改变就可用于同样类型的新模块。接线如图 3-20 所示。



图 3-20　信号模块的接线

系统提供两种端子数量的前连接器：20 针和 40 针。20 针的前连接器用于信号模块（32 通道模块除外）、功能模块和 312IFM CPU。40 针的前连接器用于 32 通道信号模块。

3.10.4　模块扩展

S7－300 PLC 是模块化的组合结构，根据应用对象的不同，可选用不同型号和不同数量的模块，并可以将这些模块安装在同一机架（导轨）或多个机架上。与 CPU312 IFM 和 CPU313 配套的模块只能安装在一个机架上。除了电源模块、CPU 模块和接口模块外，一个机架上最多只能再安装 8 个信号模块或功能模块。一个机架上模块的安装数量还受电流消耗的限制（所有模块所消耗电流总和不能超过提供的最大电流值）。

如果中央机架上没有足够的空间安装 I/O 模块，或者需要远距离安装模块，或是需要将模拟模块和数字模块分离，则需要为站点增加一个或多个扩展机架。除了带 CPU 的中央机架，S7－300 PLC 最多可以增加 3 个扩展机架。其机架扩展如图 3-21 所示。中央处理单元总是在 0 号机架的 2 号槽位上，1 号槽安装电源模块，3 号槽总是安装接口模块。4 号槽至 11 号槽，可自由分配信号模块、功能模块和通信模块。需要注意的是，槽位号是相对的，每一机架的导轨并不存在物理的槽位。例如，在机架不需要扩展机架时，CPU 模块和 4 号槽的模块是靠在一起的。此时 3 号槽位仍然被实际上并不存在的接口模块占用。

每个机架最多只能安装 8 个信号模块 SM、功能模块 FM 或通信处理模块 CP，它们安装在 4～11 号槽。

（1）只有一个扩展机架的组态

一个 S7－300 PLC 站最多可以有 4 个机架，0 号机架是主机架，1～3 号机架是扩展机架。

如果只有 1 个扩展机架，可以使用价格便宜的 IM365 接口模块对，它由两个接口模块和连接它们的 1 m 长的电缆组成。组态是将两个 IM365 模块分别插到主机架和扩展机架的第 3 槽，机架之间的连线是自动生成的。IM365 不提供直接 5 V 电源，此时，在两个机架上直流 5 V 的总电流耗量限在 1.2 A 之内。由于 IM365 不能给扩展机架提供通信总线，扩展机架上只能安装信号模块，不能安装通信处理器模块（CP）和功能模块（FM）。与使用 IM360 和 IM361 的方案相比，IM365 的价格低，使用方便，只有两个机架时应优先采用。

（2）1～3 个扩展机架的组态

中央机架使用 IM360，扩展机架使用 IM361，最多可以增加 3 个扩展机架。各相邻机架之间的电缆最长为 10 m。每个 IM361 需要接外部的 DC24 V 电源，给本扩展机架上的所有模块供电。IM360/361 有通信总线，除 CPU 和 IM360 之外的模块都可以安装在扩展机架上。用于发送的接口模块 IM360，装在 0 号机架 3 号槽。具有接受功能的接口模块 IM361，用于

S7－300 PLC 的机架 1 到机架 3 的扩展，通过连接电缆把数据从 IM360 接收到 IM361 或者从一个 IM361 传到另一个 IM361。IM361 和 IM361 之间的最大距离也为 10 m。IM361 不仅提供数据传输功能，还将 24 V 直流电压转换为 5V 直流电压，给所在机架的背板总线提供直流 5 V 电源，供电电流不超过 1. 2 A，CPU312IFM 不超过 0. 8 A。所以，每个机架所能安装的模块数量除了不能大于 8 块外，还要受到背板总线 5 V 供电电流的限制，即每个机架上各模块消耗的 5 V 电流之和应小于该机架最大的供电电流。

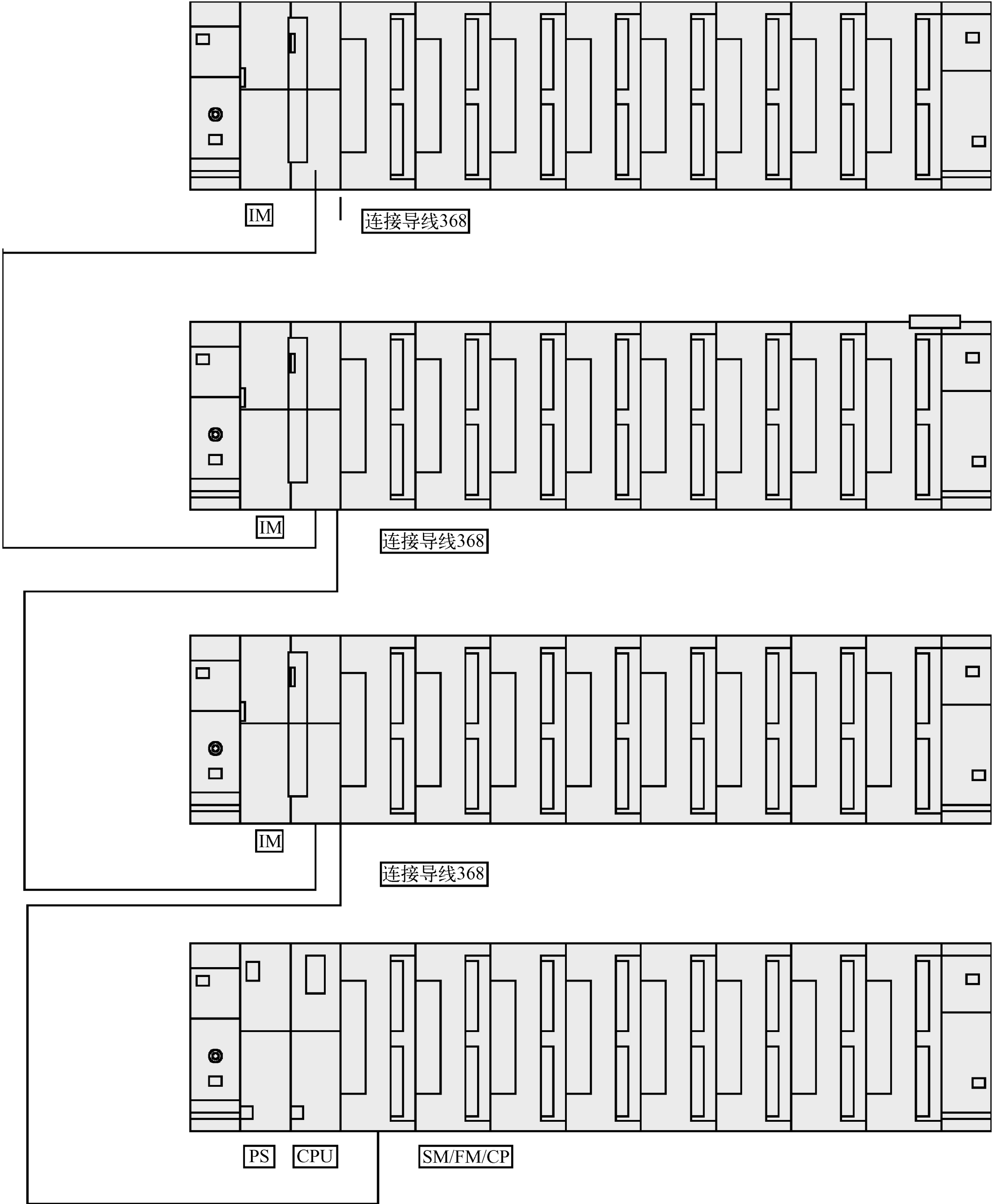


图 3-21　多机架的 S7－300 PLC 架构

3.11　S7 -300 PLC 模块编址

在进行 PLC 程序设计时，必须先确定 PLC 组成系统各 I/O 点的地址以及所用到的其他存储器（如位存储器、定时器、计数器等）的地址。PLC 通常采用两种编址方法，即绝对地址法和符号地址法。绝对地址法又有两种，即基于模块槽位的默认地址法和面向用户自定义地址的方法。

3.11.1　数字量 I/O 编址

S7 -300 PLC 的数字量（开关量）I/O 点地址由地址标示符、地址的字节部分和位部分组成，1 个字节由 0 ~7 这 8 位组成。例如 I3.2 是一个数字量输入点的地址，小数点前面的 3 是地址的字节部分，小数点后面的 2 表示它是字节中的第 2 位。I3.0 ~ I3.7 组成一个输入字节 IB3。

S7 -300 PLC 的信号模块的字节地址与模块所在的机架号和槽号有关。从 0 号字节开始，给每个数字量信号模块分配 4B（4 字节）的地址，相当于 32 个 I/O 点。M 号机架（$M=0\sim3$）的 N 号槽（$N=4\sim11$）的数字量信号的起始字节地址为 $32\times M+(N-4)\times4$。图 3-22 为 S7 -300 PLC 数字量模板的默认起始地址。

	1　2	3	4	5	6	7	8	9	10	11
机架3	PS	IM（接收）	96.0 To 99.7	100.0 To 103.7	104.0 To 107.7	108.0 To 111.7	112.0 To 115.7	116.0 To 119.7	120.0 To 123.7	124.0 To 127.7
机架2	PS	IM（接收）	64.0 To 67.7	68.0 To 71.7	72.0 To 75.7	76.0 To 79.7	80 To 83.7	84.0 To 87.7	88.0 To 91.7	92.0 To 95.7
机架1	PS	IM（接收）	32.0 To 35.7	36.0 To 39.7	40.0 To 43.7	44.0 To 47.7	48.0 To 51.7	52.0 To 55.7	56.0 To 59.7	60.0 To 63.7
机架0	PS CPU	IM（发送）	0.0 To 3.7	4.0 To 7.7	8.0 To 11.7	12.0 To 15.7	16.0 To 19.7	20.0 To 23.7	24.0 To 27.7	28.0 To 31.7

图 3-22　S7 -300 PLC 数字量模板的默认地址

说明：表中的地址范围与模板性质无关。比如在 4 号槽插入 32 点输入模板，则该板的地址为 I0.0 ~ I3.7，如插入的是 32 点输出模板，则该板的地址为 Q0.0 ~ Q3.7。如果插入的不是 32 位模板而是 16 位模板，则地址 2.0 ~3.7 自动丢失，但这些丢失的 I/O 地址仍可作为中间继电器使用。比如在 4 号槽装入一块 DI16 点的模块，则其所占用的默认地址是 I0.0 ~ I1.7；如果在 5 号槽装入一块 DO16 点的模块，则其所占用的默认地址是 Q4.0 ~ Q5.7；如果所插入的模板是开关量 I/O 模板，则 I/O 地址分别计算，比如在 4 号槽插入一个 8 点的数字输入/输出模板，则输入/输出地址分别为 I0.0 ~ I0.7 及 Q0.0 ~ Q 0.7。如果在 6 号槽装入一块 DI16/DO16 点的混合模块，则其 16 位数字量输入所占用的默认地址是 I8.0 ~ I9.7，16 位数字量输出所占用的默认地址是 QI8.0 ~ Q9.7。

除了开关量地址方式外，S7 -300 PLC 还可以使用字节（B）、字（W）或双字（D）地

址方式。如：IB4 表示由 I4.0 ~ I 4.7 共 8 位组成的一个字节的数据；IW8 表示由 IB8 及 IB9 两个字节共 16 位组成的字的内容；QD12 则表示由输出字节 QB12、13、14、15 所组成的 32 位数据。详细的访问方式在指令系统中还要进一步学习。

3.11.2　模拟量 I/O 编址

模拟量模块以通道为单位，一个通道占 1 B 或 2 B 的地址。S7 - 300 的模拟量模块的字节地址为 B256 ~ B767。一个模拟量模块最多有 8 个通道，从 256 号字节开始，S7 - 300 PLC 给每个模拟量模块分配 16 B（8 个字节）的地址。*M* 号机架的 *N* 号槽的模拟量模块的起始字节地址为 $128 \times M + (N - 4) \times 16 + 256$。图 3-23 为 S7 - 300 模拟量模板的默认起始地址。

机架3	PS		IM（接收）	640 To 654	656 To 670	672 To 686	688 To 702	704 To 718	720 To 734	736 To 750	752 To 766
机架2	PS		IM（接收）	512 To 526	528 To 542	544 To 558	560 To 574	576 To 590	592 To 606	608 To 622	624 To 638
机架1	PS		IM（接收）	384 To 398	400 To 414	416 To 430	432 To 446	448 To 462	464 To 478	480 To 494	496 To 510
机架0	PS	CPU	IM（发送）	256 To 270	272 To 286	288 To 302	304 To 318	320 To 334	336 To 350	352 To 366	368 To 382
	1	2	3	4	5	6	7	8	9	10	11

图 3-23　S7 - 300 PLC 模拟量模板的默认地址

说明：在第 4 插槽上的模拟量输入/输出地址为 256，每个模拟量模板自动按 16 B 的地址分配。每个模拟量占用 2 B，所以在模拟量地址中只有偶数。模拟量输入地址的标识符是 PIW；模拟量输出地址的标识符是 PQW，如在第 5 插槽上插入模拟量输入模板，则该模板的第 1 个通道的地址是 PIW272，如果第 7 插槽上是模拟量输出模板，则该模板的第 2 个通道地址是 PQW306。特别注意对于型号为紧凑型的 CPU（CPU31xC 或 CPU31xIFM），其 CPU 上集成的 I/O 通道地址占用了第三排扩展机架的最后一个模块的地址，即数字量占用字节 124 ~ 127，模拟量占用 752 ~ 767，所以紧凑型的 CPU 只能扩展 31 个 I/O 模块。

以上介绍了默认地址编写法。该方法的缺点是：槽位的地址不能充分利用，会造成地址丢失，使地址不连续，出现空隙，造成浪费。丢失的地址不能再作 I/O 地址用，但可作为内部中间继电器使用，用面向用户编址的方法可弥补这一缺点。

3.11.3　面向用户编址

面向用户的编址方法就是用户自定义地址的方法，用户可以通过 STEP 7 软件，设置任何所选模块的地址。S7 - 300 系列 PLC 只有 CPU315、CPU315 - 2DP、CPU316 - 2DP、CPU318 - 2DP 以及 CPU31xC 支持面向用户的编址，其他 CPU 型号的 PLC 不能采用此种编址方法。用户自定义模块地址设置方法如图 3-24 所示。定义模块的起始地址后，所有其他模块的地址都基于这个起始地址。注意：取消“系统默认”前面的对号，可以由用户自定义 I/O 地址，建议采用系统默认地址。



图 3-24　用户自定义模块地址设置

面向用户编址的优点。

① 可使模块之间不会出现地址的空隙，编址区域可充分利用。

② 当生成标准软件时，可编制独立的、不依赖于 S7-300 PLC 硬件组态的地址程序，可使软、硬件设计分开进行。

习题

3-1　填空

（1）S7-300 PLC 每个机架最多只能安装________个信号模块、功能模块或通信处理器模块。最多可以使用________个扩展机架。电源模块在中央机架最________边的 1 号槽，CPU 模块只能在________号槽，接口模块只能在________号槽。

（2）S7-300 PLC 中央机架的 9 号槽的 16 点数字量输出模块的字节地址为______________。11 号槽的 32 点数字输入模块的字节地址为________________。7 号槽的 6AI/8AO 模块的模拟量输入字的地址为________________，模拟量输出字的地址为________________。

3-2　RAM 与 FEPROM 各有什么特点？

3-3　S7-300 系列 PLC 主要有哪几类模块构成？

3-4　装载存储器和工作存储器各有什么作用，它们的区别是什么？

3-5　信号模块是哪些模块的总称？

3-6　数字量输入模块有哪几种类型？它们各有什么特点？

3-7　交流数字量输入模块与直流数字量输入模块分别适用于什么场合？

3-8　简述 RUN 方式和 RUN-P 方式有何区别。

3-9　简述复位存储器操作方法。

3-10　S7-300 系列 PLC 的扩展模块主要有哪几类？

3-11　一个控制系统需要 15 点数字量输入、24 点数字量输出、10 点模拟量输入和 3 点模拟量输出，试选择合适的输入/输出模块，并分配 I/O 地址。

第 4 章　编程软件 STEP 7 的介绍与使用

4.1　STEP 7 概述

STEP 7 是一种用于对 SIMATIC 可编程逻辑控制器进行组态和编程的标准软件包。它是 SIMATIC 工业软件的一部分。STEP 7 是一个强大的工程工具，用于整个项目流程的设计。从实施的计划配置、实施模块测试、集成测试调试到运行维护阶段，都需要不同功能的工程工具。STEP 7 工程工具可以满足整个项目流程的各种功能要求：CAD/CAE 支持、硬件组态、网络组态、仿真、过程诊断等。

STEP 7 标准软件包有下列各种版本。

STEP 7 Micro/DOS 和 STEP 7Micro/Win，用于 SIMATIC S7 －200 PLC 上的简化版单机应用程序。

STEP 7 应用在 SIMATIC S7 －300/400 PLC、SIMATIC M7 －300/400 PLC 以及 SIMATIC C7 上，具有更广泛的功能。

① 可作为 SIMATIC 工业软件产品的一个扩展选项包。

② 为功能模块和通信处理器分配参数。

③ 强制模式与多值计算模式。

④ 全局数据通信。

⑤ 使用通信功能块进行的事件驱动数据传送。

⑥ 组态连接。

STEP 7 标准组件由 SIMATIC 管理器、符号编辑器、硬件诊断、硬件组态、网络组态、多语言的用户程序编辑六部分功能组件所组成。

（1） SIMATIC 管理器

SIMATIC Manager 可以集中管理一个自动化项目的所有数据，可以分布式地读/写各个项目的用户数据。其他的工具都可以在 SIMATIC 管理器中根据需要而启动。

（2） Symbol Editor（符号编辑器）

使用 Symbol Editor（符号编辑器），可以管理所有的共享符号。其具有以下功能：可以为过程 I/O 信号、位存储和块设定符号名和注释；为符号分类；导入/导出功能可以使 STEP7 生成的符号表供其他的 Windows 工具使用。

（3） 诊断硬件

硬件诊断功能可以向用户提供可编程序控制器的状态概况。可以显示符号，指示每个模块板是否有故障。双击故障模板，可以显示故障的有关信息。

（4） 硬件组态

硬件组态工具可以为自动化项目的硬件进行组态和参数设置。可以对机架上的硬件进行配置，设置其属性。例如，设置 CPU 的启动特性和循环扫描时间的监控。通过在对话框中

提供的有效选项，系统可以屏蔽不正确的输入。

（5） NetPro（网络组态）

NetPro 工具用于组态通信网络的连接，包括网络连接的参数设置和网络各个通信设备的参数设置。选择系统集成的通信或功能块，可以轻松实现数据的传送。

拥有多种工具并不能保证得到很好的解决方法，关键在于这些工具之间能否实时地相互配合和协调。在运行一个工具的过程中，都使得项目数据库的内容不断增加或改变，我们希望这些改变在运用其他工具时能得到及时的更新或反映。STEP 7 就具有这种特点。比如，在符号表编辑器中添加变量名，那么打开程序编辑窗口时，S7 程序中这些变量立即就以变量名的形式出现。也就是说，STEP 7 的数据管理能力较强，这使得其多个工具具有连续性的工作特点。STEP 7 的优点在后续的章节中会逐渐体现出来。

（6） 编程语言

用于 S7－300 PLC 的编程语言梯形逻辑图（Ladder Logic）、语句表（Statement List）和功能块图（Function Block Diagram）都集成在一个标准软件包中。梯形逻辑图（LAD）是 STEP 7 编程语言的图形表达方式，它的指令语法与继电器的梯形逻辑图相似。语句表（STL）是 STEP 7 编程语言的文本表达式，CPU 执行程序时按每一条指令逐条地执行。功能块图（FBD）是 STEP 7 编程语言的图形表达方式，使用与布尔代数相类似的逻辑框来表达逻辑，复合功能可用逻辑框组合形式完成。

此外，还有四种编程语言作为可选软件包使用，分别是 S7 SCL（结构化控制）编程语言；S7 Graph（顺序控制）编程语言；S7 HiGraph（状态图）编程语言；S7 CFC（连续功能图）编程语言。

4.2　安装与卸载 STEP 7

为了确保 STEP 7 软件的正常、稳定地运行，不同的版本、型号对硬件、软件的安装环境有不同的要求，下面针对汉化版的 STEP 7 V5.4 为例进行说明。在安装的过程中，必须严格按照要求进行安装；此外，STEP 7 软件在安装过程中还需要进行一系列设置，如通信接口的设置等。

4.2.1　安装系统配置

（1） STEP 7 安装的硬件要求

安装 STEP 7 对硬件的要求不仅与具体的软件版本有关，还与计算机的操作系统有关。对于 Window 2000/XP 或 Window Server 2003 操作系统来说，具体的硬件要求如下。

① 在 Window 2000/XP 专业版中安装 STEP 7 V5.4 要求计算机的配置如下。

内存：512 MB 以上，推荐为 1 GB。

CPU：主频为 600 MHz 以上。

显示设备：XGA，支持 1024×768 像素分辨率，16 位以上的深度色彩。

② 在 Window Server 2003 中安装 STEP 7 V5.4 要求计算机的配置如下。

内存：1 GB 以上。

CPU：主频 2.4 GHz 以上。

显示设备：XGA，支持 1024×768 像素分辨率，16 位以上的深度色彩。

（2）软件要求

STEP 7 V5.4 可以安装在以下操作系统。

① 微软 Windows 2000 专业版（至少 SP4）。

② 微软 Windows XP 专业版（至少 SP1 或 SP1a）。

③ 微软 Windows Server 2003 工作站（有或没有 SP1）。

IE 浏览器：IE 浏览器版本要求 6.0 或更高。

4.2.2 STEP 7 授权

为了确保 STEP 7 软件的正常使用，一套正版的 STEP 7 软件除了包括两张光盘外，还有一张软盘，用于存储软件的授权。这张软盘的内容是只读的，不能复制，每安装一个授权，软盘上的授权计数器减 1，当计数器为 0 时，就不能再用它安装任何授权了。在安装 STEP 7 时可以根据提示完成安装授权；也可以在安装时跳过，待以后再安装。

STEP 7 软件即使没有授权也可以正常使用，但是在使用过程中每隔一段时间便会弹出一个“寻找授权”的对话框，以提醒使用者安装授权。

STEP 7 软件安装后，打开 Automation License Manager 软件，如图 4-1 所示，在左侧目录选中期望传输的授权所在盘，在右侧的窗口选中期望传输的授权，单击鼠标右键选择 Transfer（传输），打开传输授权对话框，选中期望的盘符即将授权传送到选择的盘符中。



图 4-1 Automation License Manager 3.0 授权管理软件

在 Automation License Manager 3.0 中可以对授权和许可证进行传输、升级、网络传输、网络共享、离线传输等操作。

为了避免丢失授权和许可密匙，需注意以下事项。

① 在格式化、压缩或恢复驱动器，安装新的操作系统之前，应将硬盘上的授权转移至软盘或其他盘中。

② 当卸载、安装、移动或升级密匙时，应先关闭任务栏可见的所有后台程序，如防病毒程序、磁盘碎片整理程序、磁盘检查程序、硬盘分区以及压缩和恢复等。

③ 使用优化软件优化系统或加载硬盘备份前，应保存授权和许可证密匙。

④ 授权和许可证密匙文件保存在隐藏目录“AX NF ZZ”中。

4.2.3 安装 STEP 7

将 STEP 7 的安装光盘插入光驱中，打开光盘，双击其中的 Setup.exe 图标，按照向导提

示进行安装。如图 4-2 所示，提示安装语言。

执行安装程序后，出现安装软件选择窗口，如图 4-3 所示，从中选择需要安装的软件。因为 STEP 7 是一个集合软件包，里面含有一系列的软件，用户可根据需要进行选择。

其中，STEP 7 V5.4 是编程软件，必须选择。

Automation License Manager 是管理编程软件许可证密钥，必须安装。

Adobe Reader 8 是阅读 PDF 格式文件的阅读器，在 STEP 7 中编写的程序是图形形式的，用户可根据具体的需要选择性安装。



图 4-2　选择 STEP7 安装语言



图 4-3　安装软件选择窗口

在图 4-3 所示窗口中完成相应设置后，单击“下一步”按钮，然后按照安装向导的提示进行操作。在正式安装软件之前，会出现图 4-4 所示的安装类型选择窗口。在此用户需要选择软件的安装类型，系统提供了典型、最小和自定义 3 种安装类型。

① 典型安装类型：安装 STEP 7 软件的所有语言、应用程序、项目示例和文档等。对于初次安装的用户来说，建议选择典型安装。

② 最小安装类型：只安装一种语言和基本的 STEP 7 程序。若要完成的控制任务比较简单，用户可选择最小安装类型方式，以节约系统资源。

③ 自定义安装类型：针对高级用户使用，根据用户自己的需求进行安装，更加灵活。

④ 在其后安装过程中还会提示用户传送密钥，如图 4-5 所示。

用户可以在安装过程中传送密钥，也可以选择安装完后再传输密钥。STEP 7 的密钥放在一张只读的软盘上，用于激活 STEP 7 软件。

在安装结束后，会弹出一个对话框，如图 4-6 所示，提示用户为存储卡设置参数。具体各项含义如下。



图 4-4　安装类型选择窗口



图 4-5　密钥传送设置

① 用户没有存储卡读卡器，则选择“None”选项，一般选择该选项。

② 如果使用内置读卡器，选择“内部变成设备接口”选项。该选项仅对西门子 PLC 专用编程器 PG 有效，对于 PC 来说是不可选的。

③ 如果用户使用的是 PC 则可选择用外部读卡器（外部存储器）。这里，用户必须定义哪个接口用于连接读卡器。

④ 在安装完成之后，用户还可以通过 STEP 7 程序组或控制面板中的“Memory Card Parameter（存储卡参数赋值）”修改这些参数设置。

在安装过程中，如图 4-7 所示会提示用户“设置 PG/PC 接口”编程设备与 PLC（PC）之间的连接有一定的原则和规律，用户根据需求进行设置。



图 4-6　存储卡参数设置



图 4-7　设置 PG/PC 接口

在“接口”栏中单击“选择”按钮，出现图 4-8 所示的“安装/删除接口”窗口，从中选择建立连接时需要安装的硬件模块。

安装 STEP 7 的注意事项如下。

① 可以用安装光盘直接安装 STEP 7，也可以将光盘中的软件复制到硬盘后再安装，但是

保存它们的文件夹的层次不能太多，且各级文件的名称不能使用中文，否则安装时可能会出现“未找到 SSF 文件”的信息，如图 4-9 所示。建议在安装软件之前关闭安全防护软件。



图 4-8　PG/PC 接口硬件模块的添加

② 如果在安装时出现“Please restart Windows before installing new programs”（安装新程序之前，请重新启动 Windows），或其他类似的英文信息，如果重新启动之后再安装软件，还会出现上述信息。可能因为 360 安全卫士这种类似软件的作用，Windows 操作系统已经注册了一个或多个保护文件，以防止被删除或重命名。具体解决方法如下。



图 4-9　SSF 文件错误

执行 Windows 的菜单命令“开始”→“运行”对话框中输入“regedit”，打开注册编辑器。选中注册表左边的文件夹“HKEY_LOCAL_MACHINE\System\Current Control Set\Control”中的“Session Manager”，删除右侧窗口中的条目“Pending File Rename Operations”，不用重新启动计算机，就可以安装软件了。可能安装每一个软件都需要做同样的操作。

③ 注意西门子自动化软件的安装顺序。用户必须先安装 STEP 7，再安装上位机组态软件 WinCC 和人机界面的组态软件 WinCC Flexible。

4.2.4　卸载 STEP 7

若用户希望安装更新版本的 STEP 7，建议首先卸载已经安装的旧版本程序。使用通常的 Window 步骤卸载 STEP 7。

① 在“控制面板（Control Panel）”中，双击“添加/删除程序（Add/Remove Programs）”图标，启动 Window 下用于安装软件的对话框。

② 在安装软件显示的项目表中，选择 STEP 7。单击“加入/删除软件（Add/Remove）”按钮。

③ 若“Remove Enable File（删除使能的文件）”对话框出现，若用户不知如何操作，

则可以单击“NO”按钮。

4.3 SIMATIC 管理器概述

SIMATIC 管理器（SIMATIC Manager）是 STEP 7 的管理平台，用于 S7 - 300/400 系列 PLC 项目组态、编程和管理的基本应用程序。STEP 7 安装完成后，单击 Windows 的“开始”→“SIMATIC”→“SIMATIC Manager”选项，或者双击桌面上图标启动 SIMATIC Manager。SIMATIC Manager 运行界面如图 4-10 所示。

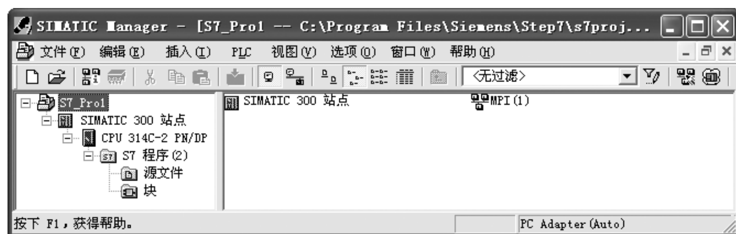


图 4-10　项目的 SIMATIC 管理器窗口

在 SIMATIC Manager 对话框内可以同时打开多个项目，所打开的每个项目均用一个项目窗口进行管理。项目窗口类似于 Windows 的资源管理器，分为左右两个视窗，左侧为项目结构视窗，显示项目的层次结构；右侧为项目对象视窗，显示左视窗对应项的内容。在右视窗内双击对象图标，可立即启动与对象相关联的编辑工具或属性窗口。

4.4 项目的管理与创建

4.4.1 STEP 7 创建项目步骤

使用 STEP 7 创建一个自动化任务解决方案的基本步骤如下。

① 根据需求设计一个自动化解决方案。

② 在 STEP 7 中创建一个项目（Project）。项目就像一个文件夹，所有数据都以分层的结构存在于其中，随时可以使用。在创建一个项目后，所有任务都在该项目下执行。其中插入工作站就是制定要使用的 PLC，如 S7 - 300 PLC、S7 - 400 PLC 等。

③ 在项目中进行硬件组态。组态硬件就是在组态表中指定控制方案所要使用的模块，以及在用户程序中以什么样的地址来访问这些模板，地址一般不用修改由程序自动生成。此外，模块的特性也可以通过修改参数进行赋值。

④ 编写程序。

⑤ 保存下载，进行在线调试，最终完成整个自动化项目。

4.4.2 项目结构

项目可用于存储为自动化任务解决方案而生成的数据和程序。这些数据被收集在一个项目下，内容如下。

① 硬件结构的组态数据及模板参数。

② 网络通信的组态数据，以及为可编程模板编制的程序。

生成一个项目的主要任务就是为编程准备这些数据。数据在一个项目中以对象的形式存储，这些对象在一个项目下按树状结构分布（项目层次），在项目窗口中各层次的显示与 Windows 资源管理器中的相似，只是对象图标不同。

项目层次的顶端结构如下。

第 1 层：项目。

第 2 层：网络、站，或 S7/M7 程序。

第 3 层：依据第二层中的对象而定。

项目窗口分成两个部分，左半部显示项目的树状结构，右半部窗口以选中的显示方式（大符号，小符号，列表，或明细数据）显示左半窗口中打开的对象中所包含的各个对象。在左半窗口单击“+”符号以显示项目的完整的树状结构。最后的结构如图 4-11 所示。



图 4-11　项目窗口组成结构

在对象层次的顶层是对象“S7 - Pro1”作为整个项目的图标。它可以用于显示项目特性并以文件夹的形式服务于网络（组态网络）、站（组态硬件），以及 S7 或 M7 程序（生成软件）。当选中项目图标时，项目中的对象显示在项目窗口的右半部分，位于对象层次（库以及项目）项部的对象在对话框中形成一个起始点，用以选择对象。

在项目窗口中，可以通过选择“offline（离线）”显示编程设备中该项目结构下已有的数据，也可以通过选择“online（在线）”，通过该项目显示可编程控制系统中已有的数据。

4.4.3　创建项目

利用 STEP 7 的 SIMATIC Manager 创建项目有两种方式：一种使用工程向导创建项目；另一种是手动创建项目。

1. 利用工程向导创建项目

① 双击桌面 STEP 7 快捷方式图标，运行 STEP 7，弹出新建项目向导；或者在 SIMATIC 管理区中选择菜单命令“File”（文件）→“New Project Wizard”（新建项目向导），打开工程向导，如图 4-12 所示。

② 单击“下一步 >”按钮，在出现对话框中选择 CPU 模块的型号，设置 CPU 在 MPI 网络中的站地址（默认值为 2），如图 4-13 所示。



图 4-12　新建项目向导对话框 1



图 4-13　新建项目向导对话框 2

③ 单击“下一步 >”按钮，在出现对话框中选择需要生成的组织块 OB，默认的是只生成作为主程序的组织块 OB1。在该对话框中还可以选择块使用的编程语言，如图 4-14 所示。

④ 单击“下一步 >”按钮，在出现的对话框的“项目名称”处修改默认的项目名称。单击“完成”按钮，开始创建项目，如图 4-15 所示。

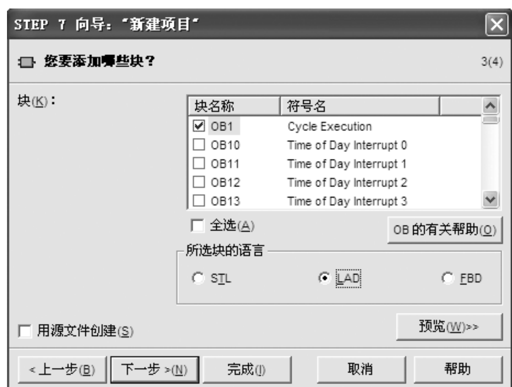


图 4-14　新建项目向导对话框 3

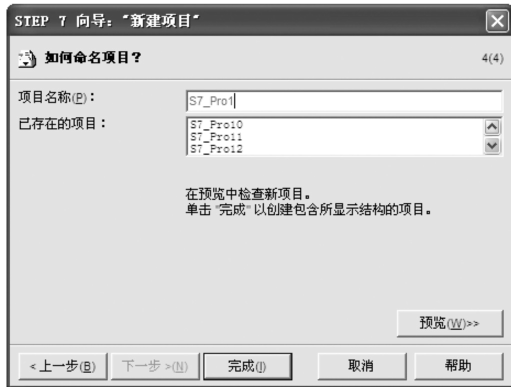


图 4-15　新建项目向导对话框 4

2. 手动创建项目

手动创建一个项目的步骤如下。

① 在 SIMATIC 管理区中选择菜单命令“文件”→“新建”，出现“新建 项目”对话框，如图 4-16 所示。



图 4-16　手动新建项目对话框

② 为项目输入名称，在“名称”文本框处输入新项目的名称，“存储位置”文本框中是默认的保存新项目的文件夹。单击“浏览”按钮，可以修改保存新项目的文件夹。最后单击“确定”按钮确认输入。注意：SIMATIC 管理器允许名字多于 8 个字符。但是，由于在项目目录中名字被截短为 8 个字符，因此一个项目名字的前 8 个字符应区别于其他的项目名称，名字不必区分大小写。

③ 用鼠标右键单击管理器中新项目的图标，在出现的快捷菜单中选择“插入新对象”命令插入一个新的 S7－300/400 站点，如图 4-17 所示。

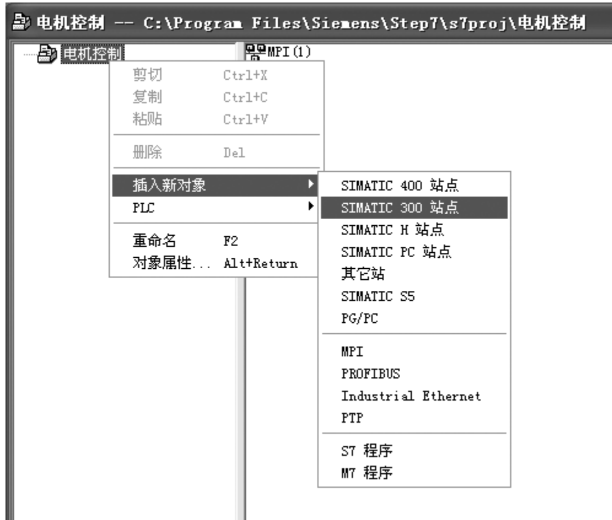


图 4-17　插入一个新的 S7－300/400 站点

或者在管理器界面菜单中选择“插入/站点/……”，如图 4－18 所示，选择一个站点即可。

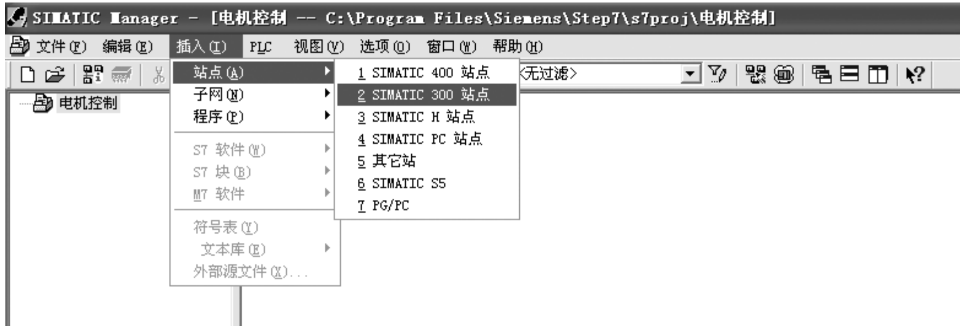


图 4-18　插入一个新的 S7－300/400 站点

④ 选中生成的站，双击右侧窗口中的“硬件”图标，在硬件组态工具 HW Config 中生成机架（导轨），将 CPU 模块、电源模块和信号模块插入机架。如果是使用工程项目向导创建的项目，机架（或导轨）和 CPU 是向导自动生成的。

4.4.4　编辑项目

（1）打开一个项目

要打开一个已存在的项目，可选择菜单命令“File”→“Open”，在随后的对话框中选中一个项目，然后，该项目窗口就打开了。

（2）复制一个项目

使用菜单命令“File”→“Save As”可以将一个项目存为另一个名字，还可以使用菜单命令“Edit”→“Copy”，复制项目的部分如站、程序、块等。

（3）删除一个项目

使用菜单命令“File”→“Delete”可删除一个项目。

如果要删除一个项目中的一部分（比如站、程序、块等），先选中项目中想要删除的部分，再使用菜单命令“Edit”→“Delete”删除一个项目中的一部分。

4.5 硬件组态

“组态”指的是在站配置机架（HW Config）窗口中对机架、模块、分布式 I/O（DP）机架以及接口模块进行排列。使用组态表示机架，就像实际的机架一样，可以在其中插入该机架相应槽对应的模块。

硬件组态的任务是在 STEP 7 的配置机架（HW Config）界面中，组态一个与实际硬件相同的硬件系统，使得软件与硬件一一对应。机架上的所有模块参数在组态过程中使用的软件设置，CPU 参数保存在系统数据（SDB）中，其他模块的参数保存在 CPU 中。

在设计一个控制系统之前，按照控制系统性质决定使用硬件及网络配置，然后在硬件组态中，定义每一个模块的参数，包括 I/O 地址、网络地址及通信波特率等参数。

4.5.1 硬件组态步骤

新建项目硬件组态有两种方法，第一种是通过“新建项目向导”完成的。在向导的引导下，新建工作变得比较轻松。本章重点介绍第二种方法：利用菜单命令“文件/新建”，建立新项目。具体步骤如下。

① 在桌面上双击 SINATIC STEP 7 软件图标，进入 STEP 7 编程软件环境。进入 SIMATIC 管理器后，执行菜单命令“文件/新建”，在打开的对话框中单击“浏览”选择文件夹，选择项目的存储位置，建立一个新项目名为“liu”（支持中文），如图 4-19 所示。

② 单击“确定”后，在 SIMATIC 管理器中，只显示出一个新建的项目名称“liu”。

在这个项目名称上按右键选择“插入行对象”，可以看到这里有多种资源供用户选择，包括 300 或 400 站点、网络和程序等。选择一个“SIMATIC 300 站点”双击右侧窗口中包含的硬件图标，打开“HM Config（硬件组态）”窗口。这时用户便可以组态硬件了。

③ 首先在右侧的硬件目录“SIMATIC 300/RACK-300”中双击“Rail（机架）”一个模拟的机架框出现在左侧的窗口中。在这个机架，用户可以配置具体的模块。首先在左侧视图机架 UR 中单击模块将要放置的位置，然后在右侧视图中双击选择的模块，或者直接将模块拖到机架上即可。如图 4-20 所示为生成机架示意图。



图 4-19 新建项目对话框

图 4-20 所示的硬件组态窗口由四部分组成。

① 左上方视图显示了当前 PLC 站中的机架 UR，用表格虚拟出了一个机架，表中的每一

行代表机架的每一个插槽。

② 左下方视图显示了机架中模块的详细信息，包括订货号、版本、地址分配等。

③ 右上方视图显示的是硬件目录，用户可以选择相应的硬件模块插入机架。

④ 右下方视图显示硬件目录中选中模块的信息，包括模块的功能、接口特性和对特殊功能的支持等。



图 4-20　生成机架示意图

需要注意的是：根据 S7 -300 PLC 硬件配置的要求，机架中的 1 号槽必须要放置电源模块。2 号槽必须放置 CPU 模块，若 CPU 模块集成了其他功能，则会在机架上多占用一定的空间。3 号槽必须放置接口模块，用于扩展更多机架，以适应更复杂的实际应用，即使没有扩展机架，该位置必须空出来。4 号槽之后可以放置其他模块，比如信号模块、通信模块。

4.5.2　参数设置

S7 -300 PLC 各种模块的参数用 STEP 7 编程软件来设置。在机架和模块详细组态信息中，双击每个模块都会弹出其属性设置窗口对话框，用户可以设置各类参数。

（1）设置 CPU 属性

双击机架上组态的 CPU 模块，弹出的对话框，包含多种选项卡。

“常规” 属性页如图 4-21 所示，包括 CPU 的基本信息和 MPI 的接口设置。单击 “属性” 按钮，如图 4-22 所示，用户可以选择建立 MPI 网络，并设置 MPI 通信速率等参数。

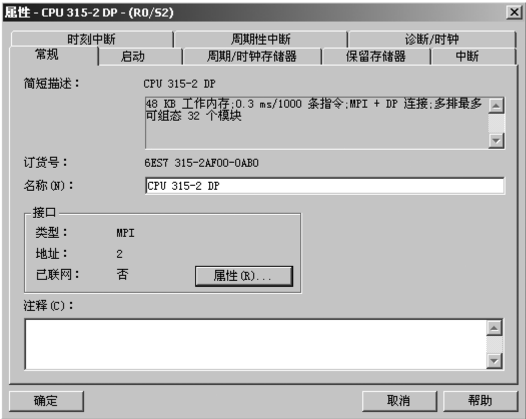


图 4-21　CPU 的 “常规” 属性窗口

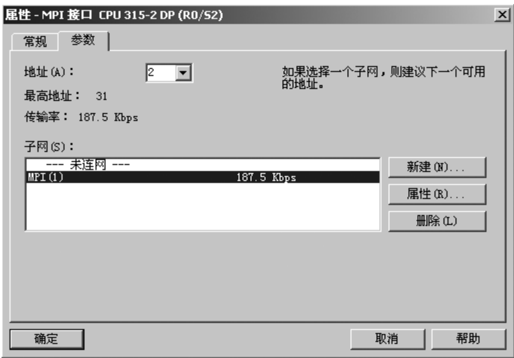


图 4-22　设置通信接口窗口

“启动”属性窗口如图4-23所示，可以设置CPU的启动特性参数。若没选中“如果预先设置的组态与实际组态不相符则启动(S)”，如果至少一个模块没有插入到组态时的指定槽位，或者插入的模块不是组态的模块式，CPU将进入STOP状态；如果选中，即使有错误，CPU也会正常启动，除了PROFIBUS-DP接口模块外，CPU不会检查I/O组态。

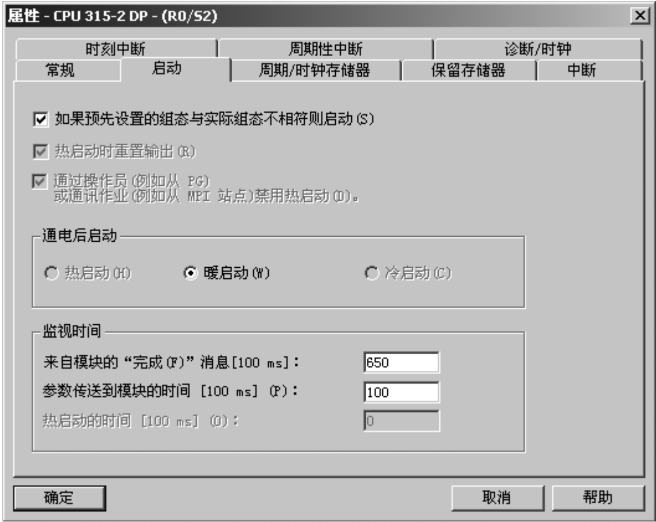


图4-23　CPU的“启动”属性窗口

其他属性的设置，用户可根据实际需要进行设置，会在后面的章节里具体介绍。

（2）设置SM的属性

双击机架组态的SM模块即可打开属性对话窗口，如图4-24所示，在“地址”选项卡中，取消“系统默认”选项后，就可以在“开始”位置修改模块的默认地址，根据用户工程的需要重新输入新地址。注意，模块的地址不能重合。

一些具有特殊功能的信号模块的参数设置方法，将在后面的章节介绍。

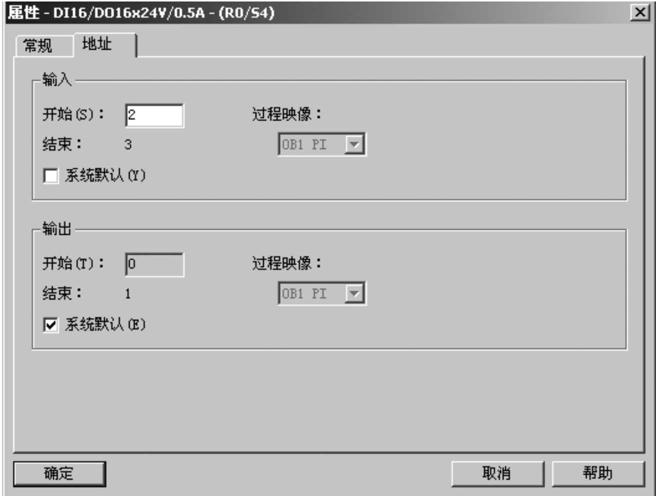


图4-24　SM模块的地址修改

4.5.3 硬件组态目录更新

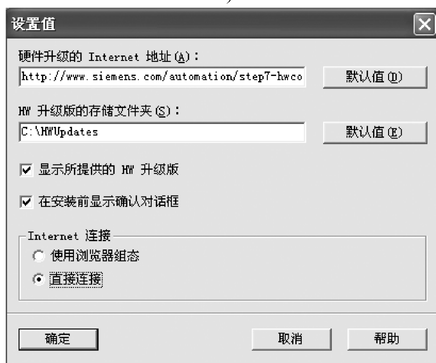
SIMATIC 是一个庞大的家族，它的每一个软硬件成员都在不断地发展。每一个 STEP 7 新版本都会支持更多、更新的硬件，但有时在硬件组态时，机架上模块的订货号必须与实际物理模块的订货号一致，而且一个型号的 PLC 模块可能有多个订货号。因此，STEP 7 提供了在线硬件更新功能。可以通过以下方法更新 STEP 7 硬件目录中的模块信息。

① 打开 STEP 7 的硬件组态窗口，在选项（Options）菜单中选择“安装 HW 更新（Install HW Updates)”命令开始硬件更新，如图 4-25 的 a 所示。第一次会提示用户设置 Internet 下载网址和更新文件保存目录，如图 4-25 的 b 所示。

② 设置完毕之后，会弹出硬件更新窗口，选择“从 Internet 下载（Download from Internet)”如果 PC 已经连接到了 Internet 上，单击“执行（Execute)”按钮就可以从网上下载最新的硬件列表，如图 4-26 所示。



a)



b)

图 4-25 STEP 7 硬件目录更新设置

a）更新菜单选择 b）更新网址设置

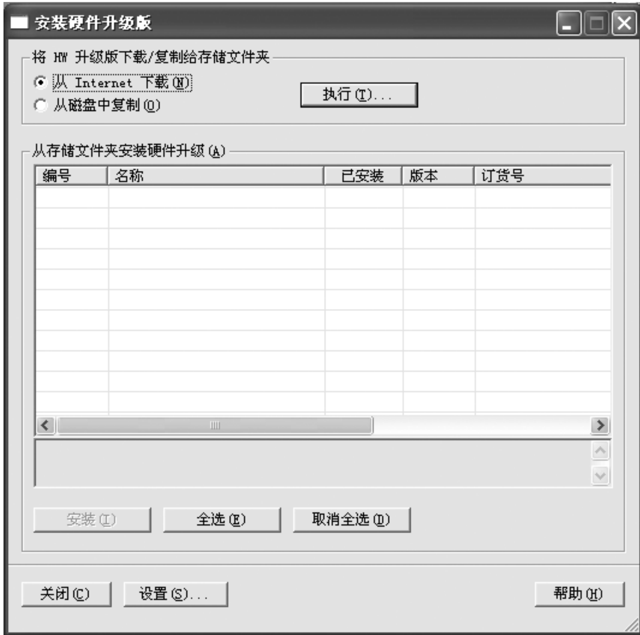


图 4-26　下载更新硬件信息

4.6　符号表

目前大部分的用户都是采用实际地址编程的方法，就如同图 4-27 中的 a 程序一样。这种方法称作按地址编程，或者称为绝对寻址方式。符号寻址允许用户有一定含义的符号地址来代替绝对地址，如图 4-27 中的 b 所示程序就是符号编程的例子。

```
A(
O     I     2.0
O     Q     3.0
)
AN    I     2.1
=     Q     3.0
```

a)

```
A(
O     "启动"
O     "电机"
)
AN    "停止"
=     "电机"
```

b)

图 4-27　绝对地址与符号地址的对比

a）绝对寻址　b）符号寻址

要实现符号编程，首先需要提前规划好所用到的内部资源，并创建一个符号表，在符号表里建立地址和符号一一对应的关系，即在符号表中为绝对地址定义具有实际意义的符号名，这样可以增强程序的可读性，简化程序的调试和维护，并且为后面的编程和维护工作能节省更多的时间。符号表如图 4-28 所示。

可以在符号表里编辑符号的元件，包括全局数据（I、Q、M、T、C、PI、PQ）和块名称（FC、FB、DB、VAT、UDT）。

STEP 7 中可以定义两类符号：全局符号和局部符号。全局符号是借助于符号编辑器来定义的，可供用户项目的所有程序块来使用。局部符号是在程序块的变量声明表中定义，只

能在该程序块中使用。

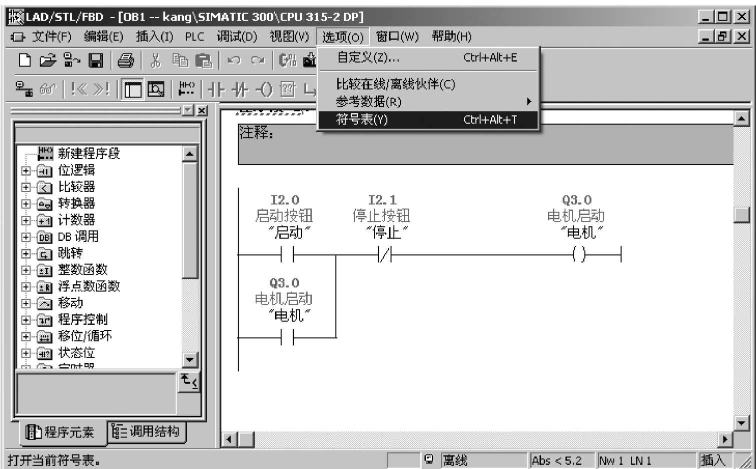
	状态	符号	地址		数据类型	注释
1		启动	I	...	BOOL	启动按钮
2		停止	I	...	BOOL	停止按钮
3		电机	Q	...	BOOL	电机启动
4						

图 4-28　符号表

4.6.1　符号表的创建

要编辑符号表，首先就要打开符号表。打开符号表的方法主要有以下几种。

① 在编辑器窗口下，选中“选项”→“符号表”，就可以打开符号表，如图 4-29 的 a 和 b 所示。



a)

	状态	符号	地址		数据类型	注释
1		启动	I	2.0	BOOL	启动按钮
2		停止	I	2.1	BOOL	停止按钮
3		电机	Q	3.0	BOOL	电机启动
4						

b)

图 4-29　在编辑器窗口打开符号表

a）编辑器窗口　b）打开符号表

② 在 SIMATIC Manager 窗口下，选中 S7 程序，在右侧的窗口下就会出现符号表图标，双击该图标即可打开符号表，如图 4-30 所示。

在符号表里，符号、地址和数据类型是必须填写的。注释根据需要填写，但最好填写清楚。并且在符号表里的符号和地址必须有唯一性。也就是说，一个符号只能与一个地址相对应；同理，一个地址也只能与一个符号相对应。

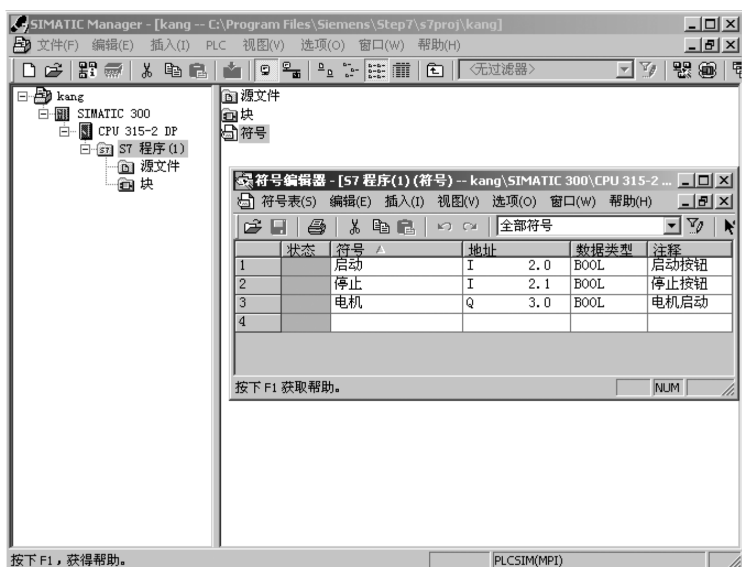


图 4-30 在 SIMATIC Manager 窗口下打开符号表

4.6.2 符号表的管理

1. 符号表的排序和过滤

一个实际应用的程序的符号表，往往有几百个到几千个符号。为了查询和修改的方便，可以进行如下操作。

（1）排序

可以按符号或地址的升/降序来排列符号表。需要排序时，在符号编辑器窗口下选中“视图”→“排序”即可打开排序对话框，如图 4-31 所示。



图 4-31 排序窗口

（2）用过滤缩小符号表显示的范围

可以利用过滤器功能，缩小符号表显示的范围，以方便寻找。在符号编辑器窗口下选中“视图”→“过滤器”，就可以打开过滤器对话窗口，如图 4-32 所示。

2. 查找和替换

可以利用查找和替换功能直接找到目标并进行修改。在符号编辑器窗口下选中“编辑”→“查找和替换”即可打开查找替换对话框，如图 4-33 所示。



图 4-32　过滤器窗口

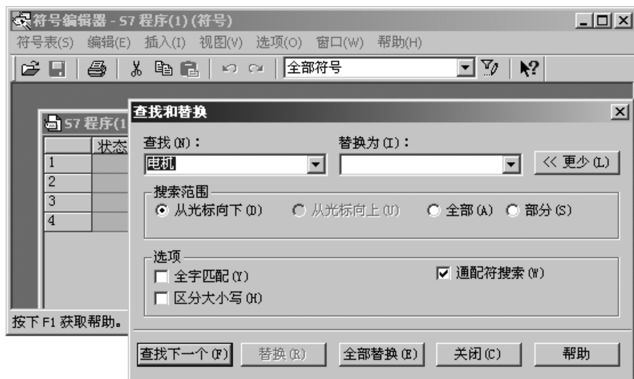


图 4-33　查找/替换窗口

3. 符号表的导入和导出

符号表往往是由做文档工作的人输入的，为了工作方便，符号表可以在别的文件格式下输入和储存。为此，STEP 7 具有符号表的导入和导出功能。

在符号编辑器窗口下，选中“符号表”→“导出”，就可以打开导出对话框，如图 4-34 所示。



图 4-34　符号表的导出

同上，在符号编辑器窗口下，选中“符号表”→“导入”，就可以打开导入对话框，如图 4-35 所示。



图 4-35　符号表的导入

4. 符号表的便利

有了符号表，程序编辑时就有很多方便。在程序编辑器窗口下，激活窗口—显示—符号表达式，程序中的地址就会变成相应的符号。激活视图—显示—符号信息，就会显示每个段内地址和符号的对应关系。

如果在符号表里有遗漏，某些地址没有分配符号，用鼠标右键单击该地址，选择“编辑符号表”，就可以补充编辑，如图 4-36 所示。

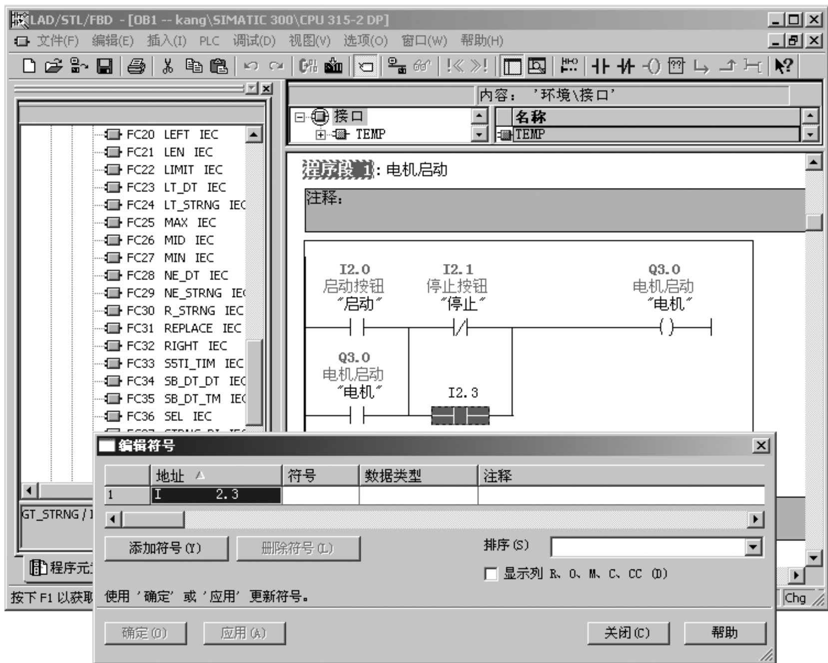


图 4-36　符号的补充编辑

4.7 逻辑块的生成

4.7.1 逻辑块的组成

逻辑块包括组织块（OB）、功能块（FB）、功能（FC）、系统功能块（SFB）和系统功能（SFC）。逻辑块由变量声明表、程序指令和块属性组成。当用户编程时，必须编辑下列3部分。

① 变量声明表：在变量声明表中，用户可以设置变量的各种参数，例如变量的名称、数据类型、地址和注释等。

② 程序指令：在程序指令部分，用户编写能被PLC执行的指令代码，可以用梯形图（LAD）、功能块图（FBD）或语句表（STL）来生成程序段。

③ 块属性：块属性中有块的信息，例如有系统自动输入的时间标记和存放块的路径。此外，用户可以输入块的名称、符号名、版本号和块的作者等。

4.7.2 程序编辑器的设置

进入程序编辑器后执行菜单命令“选项”→“自定义”打开自定义对话框，即可出现图4-37所示的对话框。

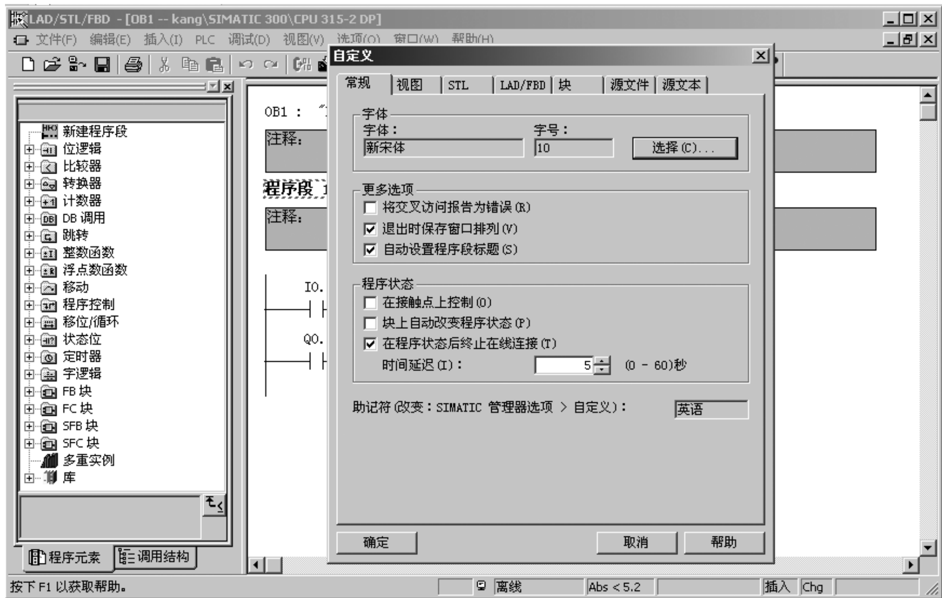


图4-37 编辑器的设置

对话框有7个选项，用户可以根据自己的需要而设置相应的参数。下面介绍一些常用的设置。

① 在“常规”选项卡的“字体”区单击按钮“选择”，可以设置编辑器使用的字体和字符的大小。

② 在“LAD/FDB”（梯形图/功能块图）选项卡可以设置地址域宽度（即触点或线圈所占的字符数）、使用二维或三维图形、线条的粗细和元件的颜色等。

③ 在“STL”（语句表）选项卡可以设置程序状态监控时显示的内容。

④ 在“块”选项卡，可以选择生成功能块时是否同时生成参考数据、功能块是否有多重背景功能，还可以选择生成块时使用的编程语言。

4.7.3 输入程序的方式

根据生成程序时选用的编程语言，输入程序的方式可以分为增量输入模式和源代码模式。

（1）增量编辑器

增量编辑器适用梯形图（LAD）、功能块图（FBD）或语句表（STL）以及 S7 Graph 等编程语言，这种编程模式适合初学者。编辑器对输入的每一行或每个元素立即进行语法检查。只有改正了指出的错误才能完成当前的输入，检查通过的输入经过自动编译后保存到用户程序中。

（2）源代码编辑器

源代码（文本）编辑器适用于语句表、S7 SCL、S7 HiGraph 编程语言。在源代码编辑器中，是用源文件（文本文件）的形式生成和编辑用户程序，再将该文件编译成各类程序块。这种编程方式又称为自由编辑方式，可以快速输入程序。

文本文件（源文件）存在项目中 S7 Program 下的“Source file”文件夹中，例如：STL source file 或 SCL source file。一个源文件可包含一个块或多个块的程序代码。用文本编辑器、STL、SCL 编程可生成 OB、FB、FC、DB 及 UDT（用户定义数据类型）的代码，也可以生成整个用户程序。CPU 的所有程序（即所有的块）可包含在一个文本文件中。当编译源文件时，就会生成各种类型的块，并在用户程序中存储起来。

在文件中使用的任何符号必须在编译之前加以定义。否则在编译的过程中，由编译器报告错误。为了顺利地通过编译，在编程语言中使用专门的句法是非常重要的。只有当选择了相容性检查命令或将源文件编译成程序块时，才运行句法检查功能。

用右键单击 SIMATIC 管理器中的“源文件”图标，执行快捷菜单命令“插入新对象”，可以生成一个新的 STL 源文件，或者插入用其他文本编辑器创建的外部源文件。

可以将块和源文件进行互相转换，其方法如下。

（1）将已生成的块转换为源文件

打开某个块，执行菜单命令“File”→“Generate Source…”（生成源文件），在出现的“New”对话框中，可以输入源文件的名称，改变保存源文件的文件夹。单击“确定”按钮，在出现的图 4-38 所示的“生成源文件”对话框中选择要转换为源文件的块，单击“确定”按钮后，选择的块被自动转换为一个源文件。

（2）将源文件编译为块

用右键单击要编译的源文件，执行出现的快捷菜单中的“Compile”（编译命令），可以将源文件转换为块，并保存在块文件夹中。如果源文件使用了符号地址，应保证这些符号地址已经在符号表中定义。

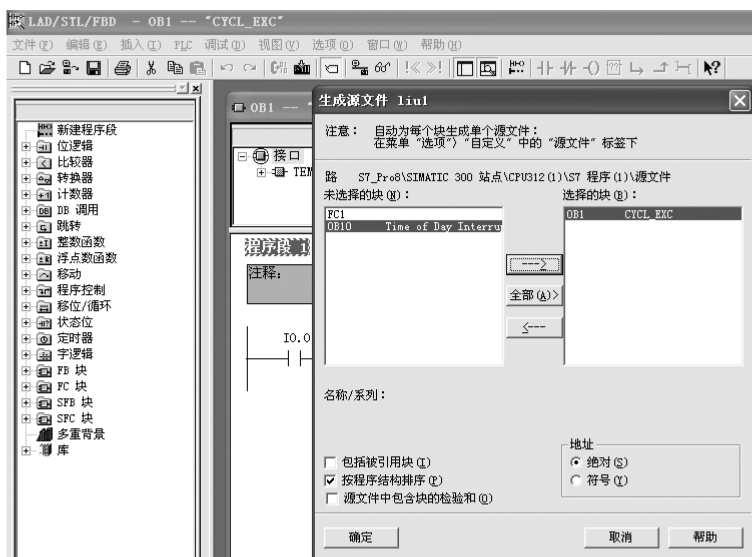


图 4-38　块转换为源文件对话框

4.7.4　程序块上锁

在 S7 Program Sources 找到生成的源文件，双击将其打开，在第 4 行中加入 "Know_How_Protect"，然后编译，无误后存盘。这样相应的块就被保护住，如果想去掉保护，在源文件中去掉 "Know_How_Protect" 编译存盘即可。

注意：千万不要丢失或删除源文件，否则程序被保护，用户可以另存到其他目录中，或 Export Source 到硬盘中，再删除源文件，这样别人只能看到未保护的块。

具体步骤如下。

① 在 STEP 7 中打开要加密的程序块，单击菜单 "文件/生成源文件（File/Generate source）"，生成要加密保护的程序块的原代码文件。

② 关闭程序块，在项目管理器中打开 "源文件（source）"，打开生成的源文件。

③ 在程序块的声明部分，TITLE 行下面的一行中输入 "Know_How_Protect"，如图 4-39 所示。

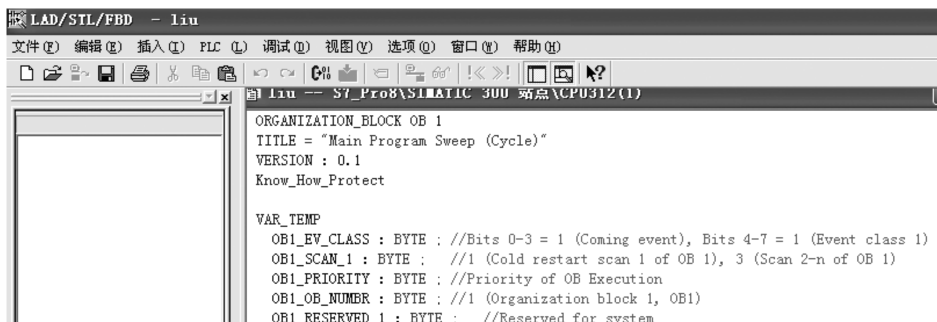


图 4-39　程序块上锁设置窗口

④ 执行菜单命令“文件/保存（File/Save）”然后“文件/编译（File/Compile）”。

⑤ 编译成功后，可在“块”中看到加锁的程序块。

⑥ 解除密码，打开程序块相应的“源文件”，删除文件中的“Know_How_Protect”，然后存盘并编译该 source 文件，即可取消对程序块的加密保护。

注意：如果没有 STL source 源文件，是无法对已经加密的程序块进行编辑的。

4.7.5 程序编辑器

1. 启动程序编辑器

在 SIMATIC Manager 窗口下双击要编辑的块的图标（比如 OB1）就可以打开编辑器窗口，如图 4-40 所示。

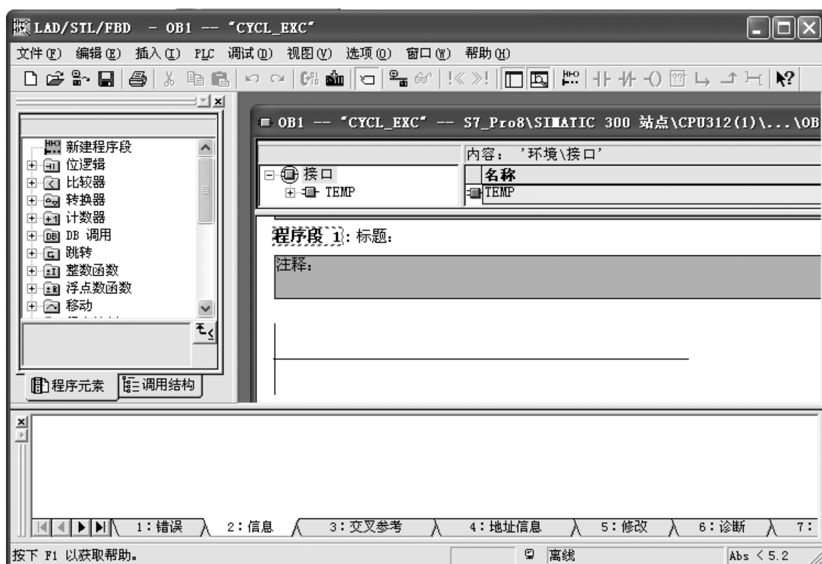


图 4-40 程序编辑器窗口

编辑器下的块由变量表和程序区两部分组成。变量声明表的用途将在以后的章节里面具体介绍。用户在当前使用中可以把变量声明表下端的线拉上去，先把变量表隐藏起来，方便用户程序的编写工作。

2. 选择编程语言

STEP 7 中有 3 种基本编程语言：梯形图（LAD）、语句表（STL）和功能块图（FBD）。还有 4 种作为可选软件包的编程语言：S7 SCL（结构化控制）、S7 GRAPH（顺序功能图）、S7 HIGRAPH（状态图）和 S7 CFC（连续功能图）。进入编辑器后，还可以再次选择编程语言，如图 4-41 所示。

如果选择 LAD 或者 FBD，则编辑时常用的元件会出现在工具条中。

LAD 工具条：

FBD 工具条：

用单击或者拖拉方法可将元件插入到光标所在的位置。工具条中没有的元件可以通过单击“总揽开/关”图标展开详细的编程元件表获得。编程软件表如图 4-42 所示。



图 4-41　选择编程语言



图 4-42　编程元件表

如果用 STL 语言编程，则只需用键盘将指令输入即可。

一个程序段编完之后，单击新程序段图标即可插入新段以便继续编程。

整个块编写完毕后，记住单击保存图标保存程序。

可以用“查看”菜单中的命令选择 3 种基本编程语言，梯形图（LAD）、语句表（STL）和功能块图（FBD）。程序没有错误时，可以在这 3 种语言中切换。STL 编写的某个程序段不能切换为 LAD 和 FBD 时，仍然用语句表表示。

3. 网络

程序被划分为若干个网络（Network），在梯形图中，每块独立的电路就是一个网络，每

个网络都有网络编号。如果在一个网络中放置一个以上的独立电路，编译时将会出错。

执行菜单命令“Insert”（插入）→“Network”，或双击工具条中的“New Network”（新程序段）图标，可以在用鼠标选中的当前网络下面生成一个新的网络。

每个程序段都有它的编号，例如“程序段 1”，程序段的标题区在程序段编号的右侧，程序段注释在标题的下面。注释下面的语句或图形是程序段的主体。

单击程序段标题区或程序段注释区，打开文字输入框，可以输入程序段的标题或注释，标题最多由 64 个字符组成。可以用菜单命令“查看”→“显示方式”→“注释”来显示或隐藏块注释和程序段注释。

可以用剪贴板在块内部和块之间复制和粘贴程序段，按住〈Ctrl〉键，用鼠标可以选中多个需要同时复制的程序段。

4. 打开和编辑块的属性

可以在生成块时编辑块的属性，生成块后可以在块编辑器中用菜单命令“文件”→“属性”来查看和编辑块属性。块属性使用户更容易识别生成的各种程序块，还可以对程序块加以保护，防止非法修改。

5. 程序的测试

将已编好的程序下载到 PLC 后，令 PLC 处于运行状态，就可以进行简单的程序测试了。把需要测试的块打开，单击“监视（开/关）”图标就进入了监控状态。若测试结果符合预期的目标，测试结束。若有需要修改的地方，可以立即修改。然后存盘下载，再进行测试，直到满意为止。

比如在 OB1 中编写一个电机单相起动/停止的程序，并且采用 PLCSIM 仿真。启动按钮 I0.0，是常开按钮。停止按钮 I0.1，是常闭按钮。驱动电机的接触器是 Q1.0。程序编辑并下载到 PLC 中运行，测试所写程序是否符合控制要求，如图 4-43 所示。

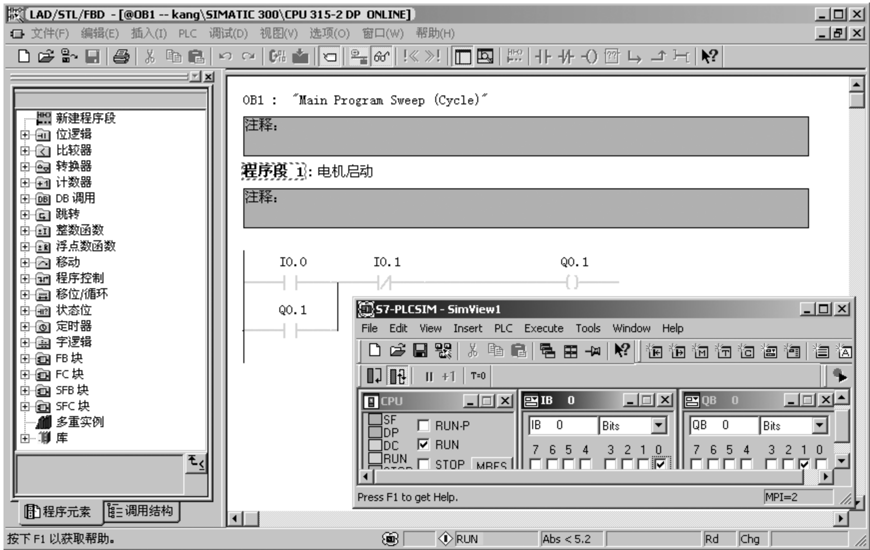


图 4-43　电机单向起/停程序测试

4.8 硬件接口与程序下载及上传

4.8.1 硬件接口

1. MPI 接口

几乎所有的 S7－300 PLC 都集成有 MPI 接口，它们既是通信接口，同时也是编程接口。因此通过 MPI 下载项目是一种比较普遍且简便的方法。

PC/MPI 适配器使用户能在编程设备（即 PC）和 PLC 之间建立数据联系，通常称它为编程电缆。一般根据与编程设备的接口不同有两种适配器，一种是 RS－232 接口，另一种是 USB 接口。适配器一端接 PLC 的 MPI 接口，另一端接编程设备的 RS－232 接口或 USB 接口，图 4-44 所示为相关产品外观，在使用前应先安装适配器驱动程序。



图 4-44　适配器、连接电缆外观图

编程电缆的价格较高，虽然如此，MPI 下载方式还是在许多不易下载的场合尤其是首次下载中发挥了重要作用。

2. PROFIBUS 接口

若在 PC 上安装通信卡，就能使编程设备和 PLC 之间通过网络进行通信。SIEMENS 提供了 CP5611 卡（PCI 卡）、CP5511 或 CP5512（PCMCIA 卡），可以将 PC 连接到 MPI 或 PROFIBUS 网络中。此时，编程电缆使用紫色的 PROFIBUS 电缆线，并需配相应的 PROFIBUS 接头。

3. 以太网接口

工业以太网通信卡，如 CP1512 卡（PCMCIA 卡）或 CP1612 卡（PCI 卡），可以将 PC 连接到以太网中。在要求不是很高的场合，也可以用普通网卡代替工业以太网通信卡。

以太网是一种非常方便而且高速的下载方式。使用以太网路径下载的前提是网络内或者下载的目的站集成有以太网口，或者通过总线扩展了 CP343－1/CP443－1 等以太网模块。一般的 PLC 自身是没有以太网接口的，但是带“PN”的 PLC 本身集成了以太网接口，使用以太网下载硬件组态或程序非常方便，比如 315 PN/DP。另外，使用以太网不仅可以下载项目，还可以在网络通信时，对网络内的各个节点同时进行监控。这是使用 MPI 下载方式无法实现的。

4. 设置 PG/PC

PG/PC Interface（PG/PC 接口）是 PG/PC 和 PLC 之间进行通信连接的接口。PG/PC 支持多种类型的接口，每种接口都需要进行相应的参数设置（如通信的波特率等）。因此，要

实现 PG/PC 和 PLC 之间的通信连接，必须正确地设置 PG/PC 接口。

在 STEP 7 安装过程中，会提示用户设置 PG/PC 接口参数。在安装完成之后，可通过以下几种方法打开 PG/PC 设置对话框。

① 在 Windows 桌面上，选择“开始”→“SIMATIC”→“设置 PG/PC 接口”命令，弹出 PG/PC 接口设置对话框。

② 在 Windows 桌面，双击“我的电脑”，再单击“控制面板”，弹出“控制面板”对话框，在该对话框中双击“Setting The PG/PC Interface”（设置 PG/PC 接口）项，弹出 PG/PC 接口设置对话框，如图 4-45 所示。



图 4-45　设置 PG/PC 接口对话框

③ 在 SIMATIC Manager 窗口中，单击菜单栏中的“选项”，再单击子菜单中的“PG/PC 接口”选项，弹出 PG/PC 接口设置对话框。

PG/PC 接口设置步骤如下。

① 将“应用程序访问点”区域设置为 S7 ONLINE（STEP 7）。

② 在“为使用的接口分配参数”区域中，选择需要的接口类型。若列表中没有需要的类型，通过单击“选择”按钮安装相应的模块或协议。

③ 选中一个接口类型，单击“属性”按钮，在弹出的对话框中对该接口参数设置。

4.8.2　程序下载及上传

（1）下载条件

① CPU 必须在允许下载的工作模式下（STOP 或 RUN-P）。建议用户在“STOP”工作模式下载。在“RUN-P”工作模式下，程序将一次下载一个模块。如果重写一个旧的 CPU 程序，可能会出现冲突。当处理循环时，CPU 就会进入 STOP 模式。

② 编程设备和 CPU 之间必须有一个连接。最常用的连接就是编程电缆了，此外还包括 PROFIBUS-DP 电缆和工业以太网的网线等，要是用户能有效访问到 PLC，不仅需要实际的物理连接，还需要设置 PG/PC 接口。

③ 用户已经编译好将要下载的程序和硬件组态。建议用户养成良好的习惯，最好能在编译完后及时保存，再下载到 PLC。这样就可以保证编程设备中的程序和 PLC 中的一致性。“保存”作用是将编程设备中当前的软硬件内容保存在编程设备的硬盘上；而“下载”的作用是将编程设备中当前的软硬件内容下载到 PLC 中。这是两个不同的概念。尤其是当用户在线调试时，用户会在线修改程序内容，这时一定要先保存程序再下载，避免下载的程序与最终保存的程序版本不一致。

（2）下载方法

在 SMATIC 管理器窗口、硬件组态窗口和“LAD/STL/FBD”窗口的工具栏上，都有下载工具按钮，而且这些窗口的菜单选项中也有下载选项“PLC/下载”，为用户提供了便捷。用户在下载时最好先下载硬件组态（确保通信的畅通），然后再下载程序。

① 在 SMATIC 管理器窗口中，首先在左视图或右视图选中要下载的对象，包括项目、PLC 站、程序块等，然后单击下载工具按钮即可。

② 在硬件组态窗口中，用户组态好的硬件后，先单击按钮编译并保存当前的硬件组态，然后单击下载按钮即可。

③ 在“LAD/STL/FBD”窗口中，单击下载按钮即可下载当前窗口编译好的程序。

注意：如果 S7 程序是硬件站的一部分，可以在块的文件中发现一个“系统数据”符号。它包含组态数据和参数分配数据，下载时确认“Do you want to load the system data?”信息，也要下载这些数据。若 CPU 处于“RUN”方式，会弹出一个信息窗口，要求自动把 CPU 切换到停机状态。在完成下载后，CPU 又自动回转为 RUN 模式。

(3) 上传条件

同“下载条件”要求一样。

(4) 上传方法

主要包括以下 3 种方法。

① 在 SMATIC 管理器窗口中，工具栏中没有上载按钮，只能通过菜单“PLC/上传到 PG”，将一个 PLC 站的内容上载到编程设备中。编程设备是直接或者通过网络连接到 PLC 上的，上载的内容包括这个 PLC 站的硬件组态和用户程序。

② 在硬件组态窗口中，通过工具栏中的上传按钮或者菜单中“PLC/上传”上传数据。这种方式会在项目中插入一个站，但是只包括硬件组态，不包括 PLC 中的用户程序。

③ 在线状态下，可以有选择地上载用户程序。在 SMATIC 管理器窗口中，单击在线按钮打开在线窗口，选中要上载的程序块，通过菜单“PLC/上传到 PG”把选项的程序块上载到编程设备中。

4.9 程序调试工具 PLCSIM

仿真软件 S7 - PLCSIM 是自动嵌套集成在 STEP 7 中的一个非常实用的软件，在 STEP 7 环境下，不用连接任何 S7 系列的 PLC（CPU 或 I/O 模块），而是通过仿真的方法来模拟 PLC 的 CPU 中用户程序的执行过程和测试用户的应用程序。可在开发阶段发现和排除错误，提高用户程序的质量和降低成本的费用。

S7 - PLCSIM 提供了简单的界面，可用编程的方法（如改变输入的通/断状态、输入值的变化）来监控和修改不同的参数，也可使用变量表（VAT）进行监控和修改变量。

4.9.1 S7 - PLCSIM 的介绍

S7 - PLCSIM V5.4 是用于 S7 - 300/400 PLC 程序仿真的 STEP 7 可选工具软件，它能够在用于编程的编程器（PG）或个人计算机（PC）上，模拟 S7 - 300/400 系列 PLC 的实际工作情况，进行程序的离线运行试验，以检验程序的正确性。

由于 S7 - PLCSIM 仿真软件具有模拟 PLC 执行用户程序全过程的功能，并可以在无任何硬件的情况下模仿实际工作状态，因此，设计者就可以在软件的设计、开发阶段发现程序中可能存在的错误与问题，验证程序的动作正确性，从而大幅度加快现场调试进度，减少了调试过程中出现故障的可能性。

通过 S7 – PLCSIM V5.4 仿真软件，可以对系统的组织块（OB）、系统功能块（SFB）、系统程序块（SFC）进行仿真。在 S7 – PLCSIM V5.4 仿真软件上，不仅可以对指令表（STL）、梯形图（LAD）、逻辑功能图（SFB）程序进行仿真，而且还可以对 S7 Graph、S7 Hi – Graph、S7 – SCL 和 CFC 程序进行仿真操作。

S7 – PLCSIM V5.4 仿真软件功能如下。

① 可以通过仿真软件运行窗口，进行 PLC 的工作模式（RUN、STOP 等）的转换，控制 PLC 的运行状态。

② 可以直接模拟生产现场，改变输入信号（I、PI）的 ON/OFF 状态，同时观察有关输出变量（Q、PQ）的状态，以监控程序的实际运行结果。在仿真的时候应注意，I/O 映像区和直接外设 I/O 是同步动作的，I/O 映像区会立即传送到外设 I/O。

③ 仿真软件可以访问模拟 PLC 的 I/O 存储器、累加器和寄存器，对模拟 PLC 的位寄存器、外围输入变量区和输出变量区以及存储的数据进行读/写操作。

④ 对定时器和计数器进行监控、修改，或通过相应的 PLC 程序使其进入自动运行状态，也可以对其进行手动复位。

⑤ S7 – PLCSIM 可以使用 PLC 的中断组织块程序测试特性，进行操作事件的记录、回放等动作，自动测试程序。

4.9.2 S7 – PLCSIM 的使用

S7 – PLCSIM 提供了一个简单的操作界面，可监控或修改程序中的参数，例如直接进行只存数字量的输入操作。当 PLC 程序在仿真 PLC 上运行时，可继续使用 STEP 7 软件中的各种功能，例如，在变量表中进行监控或修改变量。S7 – PLCSIM 的使用具体步骤如下。

① 打开 S7 – PLCSIM 。可以通过 SIMATIC 管理器中工具栏上的按钮打开/关闭仿真软件功能。如图 4–46 所示，此时系统自动装载仿真的 CPU。当 S7 – PLCSIM 运行时，所有的操作都会自动与仿真 CPU 相关联。

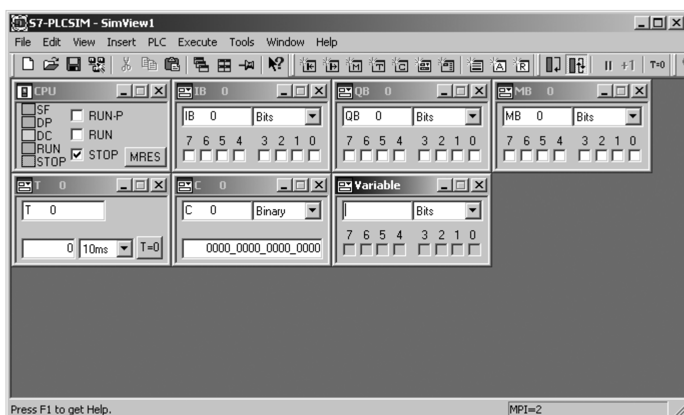


图 4–46 S7 – PLCSIM 软件界面

② 插入“View Objects”（视图对象）。通过生成视图对象（View Objects），可以访问存储区、累加器和被仿真 CPU 的配置。在视图对象上可强制和显示所有数据。执行菜单命令“Inset”或直接单击图 4–46 所示工具栏上的相应按钮，可在 S7 – PLCSIM 窗口中插入以下视

图对象。

a. Input Variable：允许访问输入（I）存储区。

b. Output Variable：允许访问输出（Q）存储区。

c. Bit Memory：允许访问位存储区（M）中的数据。

d. Timer：允许访问程序中用到的定时器。

e. Counter：允许访问程序中用到的计数器。

f. Generic：允许访问仿真 CPU 中所有存储区，包括程序使用到的数据块（DB）。

g. Vertical Bits：允许通过符号地址或绝对地址来监控或修改数据；可用于显示外部 I/O 变量（PI/PO）、I/O 映像区变量（I/O）、位存储区、数据块等。

对于插入的视图对象，可输入需要仿真的变量地址，而且可根据被监视变量的情况选择显示格式：Bits、Binary、Hex、Decimal 和 Slider：Dec（滑动条控制功能）等。变量显示“Slider：Dec”的视图如图 4-47 所示，可以滑动条控制仿真逐渐变化的值或在一定范围内变化的值。有 3 个存储区的仿真可使用这个功能：Input Variable、Output Variable、Bit Memory。

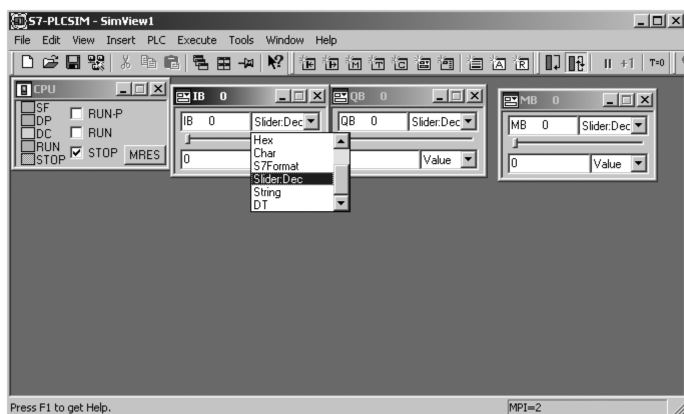


图 4-47　变量显示“Slider：Dec”

③ 下载项目到 S7 - PLCSIM 。在下载前，首先通过执行菜单命令“PLC/Power On”为仿真 PLC 上电（一般默认选项是上电）通过菜单命令“PLC/MPI Address”设置与项目中相同的 MPI 地址（一般默认 MPI 地址为 2），然后在 STEP 7 软件中单击按钮，将已经编译好的项目下载到 S7 - PLCSIM。若单击 CPU 视图中的“MRES”按钮，可清除 S7 - PLCSIM 中的内容，此时如果需要调试程序，必须重新下载程序。

④ 选择 CPU 运行的方式。执行菜单命令“Execute”→“Scan Mode”→“Singles can”，使仿真 CPU 仅执行程序一个扫描周期，然后等待开始下一次扫描；执行菜单命令“Execute”→“Scan Mode”→“Continuous can”，仿真 CPU 将会与真实 PLC 一样连续周期性地执行程序。如果用户对定时器（Timer）或计数器（Counter）进行仿真，这个功能非常有用。

⑤ 调试程序。用各个视图对象中的变量模拟实际 PLC 的 I/O 信号，用它来产生输入信号，并观察输出信号和其他存储区中内容的变化情况。模拟输入信号的方法是：用鼠标单击图 4-46 中 IB0 的第一位（即 I0.1）处的选框，则在框中出现符号“√”，表示 I0.1 为 ON；若

再次单击这个位置，则“√”消失，表示 I0.1 为 OFF。在“View Objects”中所做的改变会立即引起存储区地址中的内容发生相应变化，仿真 CPU 并不会等待扫描开始或结束才更新变换数据。执行用户程序的过程中，可检查并离线修改程序，保存后再下载，之后继续调试。

⑥ 保存文件。退出仿真软件时，可以保存仿真时生成的 LAY 文件及 PLC 文件，便于下次仿真这个项目时使用本次的各种设置。LAY 文件用于保存仿真时各视图对象的信息，如选择的数据格式等；PLC 文件用于保存仿真运行时设置的数据和动作等，包括程序、硬件组态、设置的运行模式等。

4.9.3 S7 - PLCSIM 的调试举例

以图 4-48 所示的电机起动和电机指示灯的程序为例，具体说明 S7 - PLCSIM 仿真软件的操作过程与实现效果。

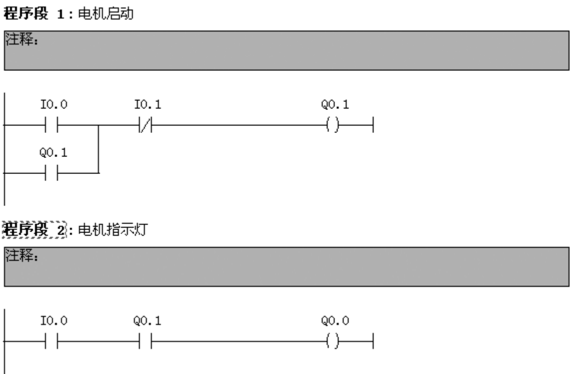


图 4-48 仿真程序例

仿真软件的操作过程如下。

① 在 STEP 7 程序编辑器中完成项目的创建、程序的编辑、硬件配置等。

② 在完成程序编辑、下载到仿真 PLC 后，打开 S7 - PLCSIM 的应用窗口。

③ 根据程序的要求，可以输入 IB0 与输出 QB0 作为仿真对象，使之在窗口显示。

④ 按照程序的要求，加入输入信号 I0.0（I0.0 = 1）。

⑤ 单击仿真 CPU 模拟面板的“RUN”按钮，使仿真 PLC 处于运行（RUN）模式。

⑥ 检查仿真输出 Q0.0、Q0.1 的结果。

从程序可知，这时 PLC 的 Q0.0、Q0.1 应输出“1”，因此，在仿真对象 QB0 中同样可以得到这一结果，实际仿真 PLC 运行后的输出 Q0.0、Q0.1 同时为“1”，如图 4-49 所示。

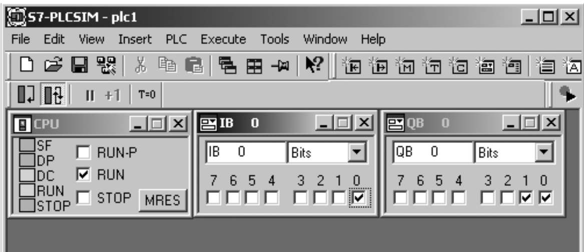


图 4-49 加入 I0.0 的仿真显示

⑦ 根据程序的设计，再加入输入信号 I0.1（I0.1 =1）。

⑧ 检查仿真输出结果 Q0.0、Q0.1 的结果。

从程序可知，这时 PLC 的 Q0.0、Q0.1 应输出“0”，因此，可以得到实际仿真 PLC 的 Q0.0、Q0.1 输出“0”，如图 4-50 所示。

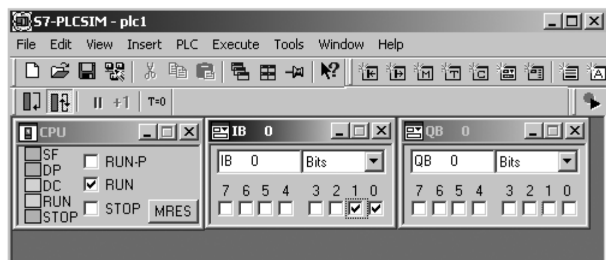


图 4-50　加入 I0.1 的仿真显示

4.9.4　PLCSIM 与真实 PLC 的差别

仿真 PLC 特有的功能如下。

① 可立即暂时停止执行用户程序，对程序状态不会有什么影响。

② 由 RUN 模式进入 STOP 模式不会改变输出状态。

③ 在视图对象中的变动立即使对应的存储区中的内容发生相应的改变，而实际 CPU 要等到扫描周期结束时才会修改存储区。

④ 可选择单次扫描或连续扫描，而实际 PLC 只能连续扫描。

⑤ 可使定时器自动运行或手动运行，可手动复位全部定时器或复位指定的定时器。

⑥ 可手动触发下列中断组织块：OB40 ~ OB47（硬件中断）、OB70（I/O 冗余错误）、OB72（CPU 冗余错误）、OB73（通信冗余错误）、OB80（时间错误）、OB82（诊断错误）、OB83（插入/拔出冗余错误）、OB85（程序顺序错误）与 OB86（机架故障）。

⑦ 对映像存储区与外设存储器的处理。如果在视图对象中改变了过程输入的值，S7 - PLCSIM 立即将它复制到外设储存器。在下一次扫描开始外设输入值被写到过程映像寄存器时，变化不会丢失，在改变过程输出值时，它被立即复制到外设输出存储区。

仿真 PLC 与实际 PLC 的区别有以下几点。

① PLCSIM 不支持写到诊断缓冲区的错误报文，例如，不能对电池失电和 EEPROM 故障仿真，但是可以对大多数 I/O 错误和程序错误仿真。

② 仿真 PLC 工作模式的改变（例如，由 RUN 转换到 STOP 模式）不会使 I/O 进入“安全状态”。

③ 仿真 PLC 不支持功能块和点对点通信。

④ 仿真 PLC 与 S7 - 400CPU 一样支持 4 个累加器。在某种情况下，仿真 PLC 上运行的程序与累加器的 S7 - 300 的程序运行可能不同。

⑤ S7 - 300 PLC 的大多数 CPU 的 I/O 是自动组态的，模块插入物理控制器后被 CPU 自动识别。仿真 PLC 没有这种自动识别功能。如果将自动识别 I/O 的 S7 - 300CPU 的程序下载到仿真 PLC，系统数据没有包括 I/O 组态。因此，在用 S7 - PLCSIM 仿真 S7 - 300 程序时，

如果想定义 CPU 支持的模块，首先必须下载硬件组态。

习题

4-1　如何利用 STEP 7 创建一个自动化项目？

4-2　练习 STEP 7 软件的安装。

4-3　为什么要进行硬件组态？如何进行 CPU 模块的参数设置？

4-4　符号表的作用是什么？

第 5 章　S7 – 300 PLC 的编程基础

5.1　IEC61131 – 3 国际标准简介

众所周知，在开发可编程序控制器（PLC）的早期阶段，由于没有一个统一的国际标准，各制造商根据自己的习惯，使用自己的编程语言，这些编程语言从内容到形式都差别很大，例如，德国的公司喜欢用功能块图和语句表语言，这与它们早期将晶体管逻辑电路用于完成控制功能有关系，而美国的公司则喜欢用梯形图语言和控制鼓（Control Drum）。梯形图语言又是从继电器控制逻辑延伸来的，至于法国公司除了用梯形图语言外，还用 GRAF-CET 语言，这一语言又特别适用于完成顺序控制的功能。

这种编程语言的不统一情况，给用户带来极大的不便，使用不同公司产品，编制的程序完全不通用，用户被迫要去熟悉不同公司的编程语言，要额外地购置不同的编程工具，要想在一个大型的工程项目中使用多家公司的产品，几乎是不可能的事。早在 20 世纪 80 年代国际电工技术委员会 IEC 的第六工作组（IEC/TC65B/WG6）就开始着手制定统一的可编程序控制器标准。并于 1993 年正式颁布了这一标准，即 IEC 61131 – 3 国际标准。

IEC（International Electrotechnical Commission）国际电工委员会是专业化国际标准化机构，负责电工、电子领域的国际标准化工作，IEC 61131 – 3 是 PLC 编程语言的国际标准，是第一个为工业自动化控制系统的软件设计提供标准化编程语言的国际标准，成为过程控制领域、分散型控制系统（DCS）、基于 PLC 的控制、运动控制和 SCADA 系统事实上的标准。该标准随着 PLC 技术的不断进步也在不断地被补充和完善。德国的 Infoteam 软件公司开发了基于 IEC61131 – 3 标准的 OpenPCS 自动化编程开发软件包，很多 PLC 厂商的编程软件都是在它的基础上开发出来的。

IEC 61131 – 3 本身只作为 PLC 的编程指导，而不是强制的规则，若 PLC 制造商希望符合这个标准，它们必须提供文件，说明其对标准的符合程度，它们还必须通过认证，确认哪些部分符合标准，哪些部分还不能满足标准。这个标准附有 62 个性能表，制造商必须填写这些表格并提供注释（例如：“完全满足”，不能“实现”，“以下部分能实现”……）。

IEC61131 – 3 国际标准得到了包括美国 AB 公司、德国西门子公司等世界知名大公司在内的众多厂家的共同推动和支持，它极大地改进了工业控制系统的编程软件质量，提高了软件开发效率；它定义的一系列图形化语言和文本语言，不仅对系统集成商和系统工程师的编程带来很大的方便，而且对最终用户同样会带来很大的便利，它在技术上的实现是高水平的，有足够的发展空间和变动余地，能很好地适应未来的工控要求。

IEC61131 – 3 国际标准正在受到越来越多的公司和厂商的重视，被越来越多的厂商采用，著名的自动化设备制造商如西门子、罗克韦尔、施耐德等公司均推出与其兼容的产品。国内近年来开发的 PLC 几乎都是符合这个标准的。IEC 61131 – 3 广泛地应用于 PLC、DCS

和工控机、“软件 PLC”、数控系统、RTU 等产品。

我国根据 IEC 的标准制定的中华人民共和国可编程序控制器（Programmable Controller）的国家标准，标准号为 GB/T15969.1－15969.4，该标准于 1995 年底发布，从 1996 年 10 月 1 日起开始实施。

5.2　S7－300 PLC 编程语言

不同生产厂家的 PLC 的编程语言通常都有较大的差异，即使同一生产厂家不同型号的 PLC 编程语言也有差异。它们的基本逻辑指令虽然较多类似，但功能指令相差较远。但如对一些基本知识能理解得较为深刻，则掌握了一种 PLC 的编程语言和编程方法，再学习另一种类型的 PLC 的编程语言和编程方法，虽不能做到“举一反三”，但还是较容易做到“触类旁通”。

STEP 7 是 S7－300/400 系列 PLC 的编程软件，所支持的 PLC 编程语言非常丰富。该软件的标准版支持以下的编程语言。

① 梯形图 LD（Ladder Diagram）：西门子简称为 LAD。

② 指令表 IL（Instruction List）：西门子称为语句表（STL）。

③ 结构文本 ST（Structured Text）：西门子称为结构化控制语言（SCL）。

④ 功能块图 FBD（Function Block Diagram）。

⑤ 顺序功能图 SFC（Sequential Function Chart）：对应于西门子的 S7 GRAPH。

STL（语句表）、LAD（梯形图）及 FBD（功能块图）3 种基本编程语言在 STEP 7 中可以相互转换。

专业版附加了对 GRAPH（顺序功能图）、SCL（结构化控制语言）、HiGraph（图形编程语言）、CFC（连续功能图）等编程语言的支持。

（1）LAD（梯形图）

梯形图是使用最多的 PLC 图形编程语言，被称为 PLC 的第一编程语言。梯形图与继电器控制电路图表达方式相仿，具有直观易懂的优点，很容易被工厂电气人员掌握，特别适用于开关量逻辑控制。梯形图结构如图 5-1 所示。

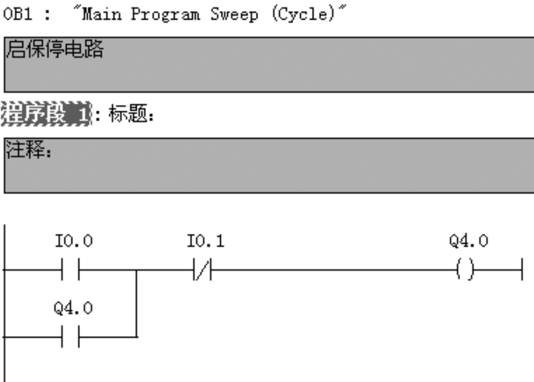


图 5-1　梯形图结构图

在 S7－300 PLC 中，触点和线圈等组成的独立电路称为网络（Network），中文版的 STEP 7 称为程序段，如图 5-2 所示。STEP 7 自动地为程序段编号。用户可以在程序段号的右侧加上程序段的标题，在程序段的下面为程序段加上注释。如果将两块独立电路（可以分开的电路）放在同一个程序段内，则将会出错。

（2）STL（语句表）

STL（语句表）是一种类似于计算机汇编语言的一种文本编程语言，由多条语句组成一个程序段。语句表可供习惯汇编语言的用户使用，在运行时间和要求的存储空间方面最优。在设计通信、数学运算等高级应用程序时建议使用语句表。语句表比较适合经验丰富的程序员使用，可以实现某些不能用梯形图或功能块图表示的功能。其形式如图 5-2 所示。

（3）FBD（功能块图）

FBD（功能块图）使用类似于布尔代数的图形逻辑符号来表示控制逻辑，一些复杂的功能用指令框表示。FBD 比较适合于有数字电路基础的编程人员使用，有数字电路基础的人很容易掌握。功能块图用类似于与门、或门的方框来表示逻辑运算关系，方框的左侧为逻辑运算的输入变量，右侧为输出变量，输入和输出端的小圆圈表示“非”运算，方框被“导线”连接在一起，信号自左向右流动。功能块图结构如图 5-3 所示，国内很少有人使用功能块图语言。

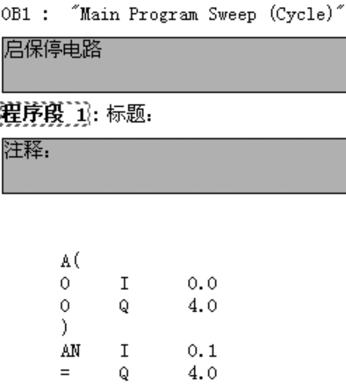


图 5-2　启保停控制语句表

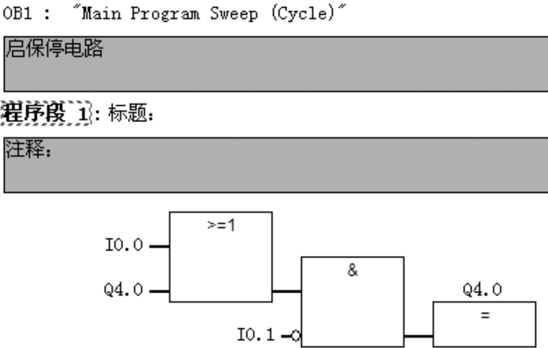


图 5-3　功能块结构图

（4）GRAPH（顺序控制）

S7 GRAPH 是 STEP 7 标准编程功能的补充，适用于顺序控制的编程。它可以清楚快速地组织和编写 S7 PLC 系统的顺序控制程序。它根据功能将控制任务分解为若干步，其顺序用图形方式显示出来并且可形成图形和文本方式的文件。同时 S7 GRAPH 还表达了顺序的结构，以方便进行编程、调试和查找故障。

（5）S7－HiGraph（图形编程语言）

图形编程语言 S7－Higraph 属于可选软件包，它允许用状态图描述生产过程，将自动控制下的机器或系统分成若干个功能单元，并为每个单元生成状态图，然后利用信息通信将功能单元组合在一起形成完整的系统。

（6）SCL（结构化控制语言）

S7－SCL（Structured Control Language，结构化控制语言）是一种类似于 PASCAL 的高级

文本编辑语言，用于 S7－300/400 和 C7 的编程，可以简化数学计算、数据管理和组织工作。S7－SCL 具有 PLC 公开的基本标准认证，符合 IEC 1131－3（结构化文本）标准。

5.3　梯形图绘制原则

PLC 梯形图的绘制应该遵循以下基本原则。

① 梯形图按元件从左到右、从上到下绘制。

② 梯形图中的触点应画在水平支路上，不应画在垂直支路上。

③ 梯形图中只出现输入继电器的触点而不出现输入继电器的线圈。

④ 外部输入、输出、内部继电器（位存储器）、定时器、计数器等器件的触点可多次重复使用。

⑤ 梯形图每一行都是从左侧母线开始，线圈接在最右侧，触点不能放在线圈的右侧。

⑥ 线圈不能直接与左侧母线相连。

⑦ 同一编号的线圈在一个程序中使用两次及以上（称为双线圈输出）容易引起误操作，应尽量避免双线圈输出。

⑧ 梯形图程序必须符合顺序执行的原则，从左到右，从上到下地执行，如不符合顺序执行的电路不能直接编程。

⑨ 在梯形图中串联触点、并联触点的使用次数没有限制，可无限次地使用。

5.4　变量与数制

5.4.1　变量

变量是保存在存储器中待处理的抽象数据，是为了识别 PLC 的输入/输出、PLC 内部的存储区域而使用的名称，可以代替物理地址在出现中使用。简单变量由一个地址和一个数据类型组成，其中，地址由一个区域标识符和一个存储地址组成。

变量分为全局变量和局部变量。变量通过声明确定相关属性，如关键字、数据类型和有效范围等。全局变量可以在 CPU 范围内被所有的程序块调用，例如在 OB（组织块）、FC（功能）、FB（功能块）中使用，在某一个程序块中赋值后，在其他的程序中可以读出，没有使用权限。局部变量只是在所属的程序块（OB、FC、FB）范围内使用，在程序块调用时有效，程序块调用完成后被释放，所以不能被其他程序块使用，本地数据区（L）中的变量为区域变量，例如每个程序块中的临时变量都属于区域变量。

5.4.2　常量和数制

常量是用于给变量预留的一个固定值，常量的特殊前缀根据其数据类型给出。不同的数据类型有着不同的数制，在 PLC 中运用比较广泛，下面介绍在 S7－300 PLC 中常用的数制。

（1）二进制数

二进制数的 1 位（bit）只能取 0 和 1 这两个不同的值，用于表示开关量的两种不同的状态，例如触点的断开和接通、线圈的通电和断电等。如果该位为 1，则表示梯形图中对应

的位编程元件的线圈“通电”，则其常开触点接通，常闭触点断开，以后称该编程元件为1状态，或称该编程元件ON（接通）。如果该位为0，则对应的编程元件的线圈和触点的状态与上述的相反，称该编程元件为0状态，或称该编程元件OFF（断电）。二进制数一般用2#表示，例如：2#0110_ 1100_ 1010_ 1000表示16位二进制常数。

（2）十六进制数

多位二进制数的书写和阅读很不方便，为了解决这一问题，可以用十六进制来取代二进制数，十六进制的16个数字是0～9和A～F（对应于十进制数的10～15），每个占二进制数的4位。在SIMATIC中用B#16#、W#16#、DW#16#分别用来表示十六进制的字节、字和双字常数，例如B#16#A5、W#16#B2A5、DW#16#BACF6983、W#16#13AF（13AFH）。在数字后面加“H”也可以表示十六进制数，如46AFH。

十六进制数的运算规则为逢16进1，例如B#16#AB+B#16#2C=B#16#D7。

（3）八进制数

八进制数的8个数字采用0～7八个数字，逢八进位。通常在数字后面加“O”来表示八进制数。

（4）BCD码

BCD码用4位二进制数表示一位十进制数，十进制数9对应的二进制数为1001。最高4位用于表示符号，负数的最高位为1，正数为0，其余3位可以取0或1，一般取1。16位BCD码的范围为-999～+999，32位BCD码的范围为-9999999～+9999999。BCD码实际上是十六进制数，但是各位之间逢十进一。296对应的BCD码为W#16#296，或2#0000 0010 1001 0110。

不同进制的数的表示方法见表5-1。

表5-1　不同进制的数的表示方法

十进制数	十六进制数	二进制数	BCD码
0	0	00000	0000 0001
1	1	00001	0000 0001
2	2	00010	0000 0010
3	3	00011	0000 0011
4	4	00100	0000 0100
5	5	00101	0000 0101
6	6	00110	0000 0110
7	7	00111	0000 0111
8	8	01000	0000 1000
9	9	01001	0000 1001
10	A	01010	0001 0000
11	B	01011	0001 0001
12	C	01100	0001 0010
13	D	01101	0001 0011
14	E	01110	0001 0100
15	F	01111	0001 0101
16	10	10000	0001 0110
17	11	10001	0001 0111

5.5 数据类型

数据类型决定数据的属性，在 SEPT 7 中，数据类型分为三大类。

① 基本数据类型。

② 由基本数据类型组合的复杂数据类型。

③ 用于传递 FB（功能块）和 FC（功能参数）的参数类型。

5.5.1 基本数据类型

基本数据类型是根据 IEC61131－3 来定义的，基本数据类型共有 12 种，其长度不超过 32 位，可以装入 S7 处理器的累加器中，可利用 STEP 7 基本指令处理。数据类型决定了数据长度、数据格式及常数表达格式等。表 5-2 给出了 STEP 7 中的基本数据类型描述及常数举例。

表 5-2 基本数据类型

数据类型	描述	位数	常数举例
BOOL	二进制位	1	1/0
BYTE	字节	8	B#16#F5
WORD	无符号字	16	W#16#69AD
INT	十进制有符号整数	16	－896
DWORD	无符号双字	32	DW#16#5968ADCF
DINT	十进制有符号双整数	32	L#231
REAL	IEEE 浮点数	32	51.3
S5TIME	S7 时间	16	S5T#2H5M50S
TIME	IEC 时间	32	T#1H6M6S
DATE	IEC 日期	16	D#2013_11_6
TIME_OF_DAY	实时时间	32	TOD#1:15:30.2
CHAR	ASCII 字符	8	‘A’

下面详细地介绍 STEP 7 的基本数据类型。

（1）位（bit）

位数据的数据类型为 BOOL（布尔）型，BOOL 的取值有 0 和 1（或 true 和 false），存储空间为 1 位。位存储单元的地址在 S7－300 中由字节地址和位地址组成，在 SEPT 7 中的表示方法为“字节地址．位地址”，例如，I2.2，其中的“I”表示区域标示符，代表的是输入（Input），字节地址为 2，位地址为 2，如图 5-4 所示。这种存取方式成为“字节．位”寻址方式。

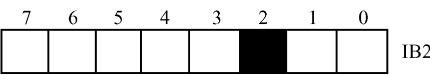


图 5-4 位数据存放

（2）字节（Byte）

一个字节由 8 个位组成，通过地址标识符 B 和表示绝对地址的一个字节来表示，数据

格式为 B#16#，B 代表 BYTE，表示数据长度为一个字节（8 位），#16#表示十六进制，取值范围为 B#16#0 ~ B#16#FF。如图 5-4，IB2 是一个字节，它的位地址由低到高为 I0. 0 ~ I0. 7，其中第 0 位为最低位（LSB），第 7 位为最高位（MSB）。

（3）字（Word）

相邻的两个字节组成 1 个字（Word），字用于表示无符号数，同样是通过地址标识符 W 和表示绝对地址的变量高字节所在的地址来表示的，字的取值范围为 W#16#0000 ~ W#16#FFFF。为了避免地址交叉，地址一般为 2 的倍数。例如：IW6 表示输入地址是 6，并且包含 IB6 和 IB7 两个字节；MW100 是由 MB100 和 MB101 组成的一个字，如图 5-5 中的 b 图所示，其中 M 为区域标识符，W 表示字。

（4）双字（Double Word）

DWORD 包含四个字节，是通过地址标识符 D 和表示绝对地址的变量高字节所在的地址来表示的，其范围 DW#16#0000_0000 ~ DW#16#FFFF_FFFF。为了避免地址交叉，地址一般为 4 的倍数。例如：ID4 表示输入地址是 4，而且包含 IB4、IB5、IB6、IB7 四个字节。字节、字和双字的关系如图 5-5 所示。

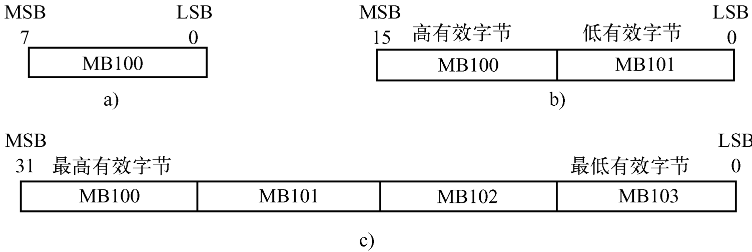


图 5-5　字节、字和双字

a）MB100　b）MW100　c）MD100

图 5-5 中需要注意以下问题。

① 以组成字 MW100 和双字 MD100 的编号最小的字节 MB100 的编号作为 MW100 和双字 MD100 的编号。

② 组成字 MW100 和双字 MD100 的编号最小的字节 MB100 为 MW100 和 MD100 的最高位字节，编号最大的字节为字和双字的最低位字节，这就是所谓的“高地址，低字节”的约定。

③ 数据类型字节、字和双字都是无符号数，它们的数值用十六进制来表示。

（5）16 位整数（INT，Integer）

整数有符号数，在进行数字运算时使用，其应用与计算机中的定义相同。整数数据类型长度为 16 位，数据格式为带符号十进制数，16 位中最高为符号位。正整数是以原码格式进行存储的，如 +786，对应的二进制码为 2#0000_0011_0001_0010，而负整数则表示为正整数的二进制补码，即对应正整数的二进制码取反后加 1，例如负整数 -786，对应的二进制码为 2#1111_1100_1110_1110。将负零（1000_0000_0000_0000）定义为 -32768 因此取值范围为 -32768 ~ 32767。其最高位为符号位，为 0 时为正数，为 1 时为负数。

（6）32 位整数（DINT，Double Integer）

双整数的数据类型长度为 32 位，数据格式为带符号十进制数，用 L#表示双整数。双整

数的二进制码与整数的换算方式一致，32 位整数最高位为符号位，其取值范围为 L# -2147483648 ~ L#2147483647。

（7）32 位浮点数

浮点数又称实数（REAL），定义同计算机中的格式。这里需要注意：PLC 输入输出值大多为整数，浮点数的运算速度比整数运算的慢。实数的数据类型长度为 32 位，是以 IEEE 浮点数格式转换为二进制数存储的，其取值范围为 $3.402823e^{+38}$ ~ $-1.1755494e^{-38}$。

在 STEP 7 中，一般并不使用二进制格式或者十六进制格式表示浮点数，而是用十进制小数来输入或显示浮点数，比如在 STEP 7 中，40.0 为浮点数，而 40 则为整数。

（8）S5 TIME（SIMATIC 时间）

S5 TIME 时间数据类型长度为 16 位，包括时基和时间常数两部分。

（9）IEC 时间（TIME）

IEC 时间数据长度为 32 位，时基为固定值 1 ms，数据类型为双整数，所表示的时间值为整数值乘以时基。格式为：

T#aaD_bbH_ccM_ddS_eeeMS

其中 aa = 天数，bb = 小时数，cc = 分钟，dd = 秒，eee = 毫秒。根据双整数的最大值为 2147483647，乘以时基 1 ms，可以算出，IEC 时间的最大值为 T#24D_20H_31M_23S_648Ms。使用 IEC 时间时，时基保持不变，且精度较高，为 1 ms。

（10）IEC 日期（DATE）

IEC 日期数据长度为 16 位，数据类型为整数，以 1 日为单位，日期从 1990 年 1 月 1 日开始，1990 年 1 月 1 日对应的整数为 0，日期每增加一天，对应的整数值加 1，如 23，对应 1990 年 1 月 22 日。IEC 日期格式为 D#_年_月_日，例如 D#1992_12_2。取值范围为 D#1990_1_1 ~ D#2163_12_31。

（11）日计时（TIME_OF_DAY）

日计时是表示一天中的 24 小时，数据长度为 32 位，数据类型为双整数，以 1 ms 为时基，取值范围为 TOD#0:0:0:0 ~ TOD#23:59:59.999。

（12）字符（CHAR）

字符数据的长度为 8 位，字符采用 ASCII 码的存储方式。

（13）常数的表示方法

L#为 32 位双整数常数，例如 L# +5。

P#为地址指针常数，例如 P#M2.0 是 M2.0 的地址。

S5T#是 16 位 S5 时间常数，格式为 S5T#aH_bM_cS_dMS。S5T#4S30MS = 4s30ms，取值范围为 S5T#0 ~ S5T#2H_46M_30S_0MS（9990 s），时间增量为 10 ms。

C#为计数器常数（BCD 码），例如 C#250。

8 位 ASCII 字符，用单引号表示，例如‘ABC’。

T#为带符号的 32 位 IEC 时间常数，例如：T#1D_12H_30M_0S_250MS。

DATE 是 IEC 日期常数，例如 D#2004-1-15。取值范围 D#1990-1-1 ~ D#2168-12-31。

TOD#是 32 位实时时间（Time of day）常数，例如 TOD#23:50:45.300。

5.5.2　复杂数据类型

复杂数据类型，就是基本数据类型的组合，只能结合共享数据块的变量声明使用。数据长度超过32位。因为数据长度超过累加器的长度，所以不可以一次性用装入指令把整个数据装入累加器中，一般利用库中的标准块（“IEC”S7程序）处理复杂数据。

复杂数据类型包括日期和时间（DATE_AND_TIME）、字符串（STRING）、数组（ARRAY）、结构（STRUCT）和用户定义类型（UDT）。

① 日期和时间：日期和时间用8个字节的BCD码来存储。第0～5号字节分别存储年、月、日、时、分和秒，毫秒存储在字节6和字节7的高4位，星期存储在字节7的低4位。例如2004年7月27日12点30分25.123秒可以表示为DT#04－07－27－12:30:25.123。

通过调用程序编辑器的文件夹“\Libraries\Standard Library\IEC Function Block”中的IEC功能，可以实现DATA_AND_TIME数据类型与基本数据之间的相互转换、日期时间的比较和加减。

② 字符串：字符串（STRING）是字符（CHAR）组成的一维数组，每个字节存储1个字符。字符串（STRING）由最多254个字符（CHAR）和2字节的头部组成。第一个字节是字符串的最大字符长度，第二个字节是字符串当前有效字符的个数，字符从第三个字节开始存储，字符串的默认长度为254B。在DB4中定义字符串Fault的长度为20个字符，它只占用从DB4.DBB20开始的22B，其初值只有4个字符Over。STRING变量中未使用的字节地址被初始化为B#16#00。

将DB4下载到仿真PLC，用变量表监控DB4.DBB20和DB4.DBB21，它们的值分别为20和4。还可以看到从DB4.DBB22开始的4个字节中字符Over的ASCII码。用户可以用语句表中的基本指令访问字符串中的字符，例如用指令“L DB4.Fault[3]”来访问字符串Fault中的第三个字符。

用户可以用标准库的IEC库中的21个FC来处理字符串变量，包括字符串与其他数据类型的转换、字符串的比较和字符串的编辑。

③ 数组：数组（ARRAY）是同一类型的数据组合而成的一个单元，数组的维数最多为6维。ARRAY后面的方括号中的数字用来定义每一维的起始元素和结束元素在该维的编号，可以取－32768～32767之间的整数。各维之间的数字用逗号隔开，每一维开始和结束的编号用两个小数点隔开。如果某一维有 n 个元素，该元素的起始元素和结束元素的编号可以采用1和 n，例如ARRAY[1..100]，ARRAY[1..2,1..3]是一个二维数组，共有6个整数元素，其结构如图5-6所示。

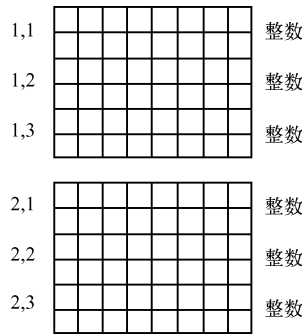


图5-6　二维数组的结构

数组元素“liu”.shu[2,1]：liu是数据块的符号名，shu是数组的名称。方括号中是数组元素的下标。如果在块的变量声明表中声明形参的类型为ARRAY，可以将整个数组而不是某些元素作为参数来传递。

④ 结构：结构（STRUCT）可以是不同类型的数据组合。用户可以用基本数据类型、复杂数据类型（包括数组和结构）和用户定义的数据类型（UDT）作为结构中的元素，不同

于数组的不能嵌套，每个结构最多允许嵌套 8 层。例如一个结构可以由数组和结构组成。用户可以把过程控制中有关的数据统一组织在一个结构中，作为一个数据单元来使用，而不是使用大量的单个的元素，为统一处理不同类型的数据或参数提供了方便。

数据块 liu 内结构 struct 的元素 ceshi 应表示为“liu”. struct. ceshi。

将结构作为参数传递时，作为形参和实参的两个结构必须有相同的数据结构，即相同数据类型的结构元素和相同的排列顺序。结构元素可在声明中定义初始值，在“Initial value”一栏中输入初始值，初始化的数据类型必须与结构元素的数据类型相一致。

结构变量从字边界开始，也就是说，起始地址为偶数字节地址。随后，每个结构元素以其声明时的顺序依次存储到存储器中，一个结构类型变量占一个字的存储空间。数据类型为 BOOL 的结构元素从最低位开始存储；数据类型为 BYTE 和 CHAR 的结构元素从偶数字节开始存储；其他数据类型的结构元素从字地址开始存储。

⑤ 用户定义数据类型：用户定义数据类型（User-Defined Data Types，UDT）是一种特殊的数据结构，用户只需要对它定义一次，定义好以后可以在用户程序中作为数据类型使用。用户可以用它产生大量的具有相同数据类型的数据块，用这些数据块来输入用于不同目的的实际数据。例如，生成用于颜料混合配方的 UDT 后，可以用它生成用于不同颜色的数据组合。用户定义数据类型由基本数据类型和复杂数据类型组成。定义好以后可以在符号表中为它指定一个符号名，使用 UDT 可以节约录入数据的时间。

5.5.3 参数类型

参数类型是为逻辑块的形式参数（简称形参）定义的数据类型，用于在调用逻辑块时传递参数。除了简单数据类型和复杂数据类型之外，还可以使用下列的参数类型。

（1）定时器（TIMER）与计数器（COUNTER）

使用参数类型 TIMER（定时器）和 COUNTER（计数器）可以在调用逻辑块时，分别将定时器和计数器的编号（例如 T3、C21）作为实参传送给块的形参。

（2）块（BLOCK）

使用参数类型 Block_FC、Block_FB、Block_DB 和 Block_SDB，可以在调用逻辑块时，分别将 FC、FB、DB 和 SDB（系统数据块）作为实参传送给块的形参。块参数类型的实参应为同类型的块的绝对地址编号（例如 FB 2）或块的符号名。

参数类型 TIMER、COUNTER、BLOCK 只能用于块的输入变量（IN）的形参。

（3）指针（POINTER）

参数类型 POINTER（指针）可以指向一个变量的地址，例如 P# M0.0 表示指向 M0.0 的指针。

（4）任意参数（ANY）

在调用逻辑块时，长度为 10B 的数据类型 ANY 指针，用于将任意的数据类型传递给声明的形参。ANY 可用于实参的数据类型未知，或实参可以使用任意数据类型的情况。ANY 指针可以用于表示一片连续的数据区，在数据通信中经常使用。例如 P#DB2. DBX50.0 BYTE 10，其表示 DB 2 中的以 50 地址为起始地址的 DBB50 ~ DBB59 共 10 个字节。

5.6　PLC 中的存储区与寄存器

5.6.1　系统存储器

PLC 的用户存储区按功能划分为不同的地址区，在程序中通过指令，可以直接访问存储于地址区的数据。S7－300/400 存储区划分、功能、访问方式及表示见表 5-3。

表 5-3　S7－300/400 PLC 存储区划分、功能及标识符

存储区名称	区 域 功 能	访问区域单位	标　识　符	最大地址范围
输入过程映像存储区（I）	在扫描循环的开始，操作系统从现场读取控制按钮、行程开关及各种开关量数字信号，并存入输入过程印象寄存器，其每一位对应数字量输入模块的输入端子	输入位 输入字节 输入字 输入双字	I IB IW ID	0～65 535.7 0～65 535 0～65 534 0～65 532
输出过程映像存储区（Q）	在扫描循环期间，逻辑运算的结果存入输出过程映像寄存器，在循环扫描结束前，操作系统从输出过程映像寄存器读出最终结果。并将其传送到数字量输出模块，直接控制 PLC 外部的指示灯、接触器等控制对象	输出位 输出字节 输出字 输出双字	Q QB QW QD	0～65 535.7 0～65 535 0～65 534 0～65 532
位存储器（M）	位存储器与 PLC 外部的对象没有任何关系，它的功能类似于继电器控制系统中的中间继电器，主要用于存储程序运算过程中的临时结果，可为编程提供无数量限制的触点，可以被驱动，但不能直接驱动任何负载。它的存在主要是为了使编程方便	存储位 存储字节 存储字 存储双字	M MB MW MD	0～255.7 0～255 0～254 0～252
外部输入（PI）	用户可以通过外部输入寄存器直接访问模拟量输入模块，以便接收来自现场的模拟量输入信号	外部输入位 外部输入字 外部输入双字	PIB PIW PID	0～65 535 0～65 534 0～65 532
外部输出（PQ）	用户可以通过外部输出寄存器直接访问模拟量输出模块，以便将模拟量输出信号送给现场的控制执行器	外部输出位 外部输出字 外部输出双字	PQB PQW PQD	0～65 535 0～65 534 0～65 532
定时器（T）	作为定时器指令使用，访问该存储区可获得定时器的剩余时间	定时器	T	0～255
计数器（C）	作为计数器指令使用，可访问存储区可获得计数器的当前值	计数器	C	0～255
数据块（DB）	数据块寄存器用于存储数据块的数据，最多可同时打开一个共享数据 DB 和一个背景数据块 DI，用 OPIN DB 指令可打开一个共享数据块，OPEN DI 指令打开背景数据块	数据位 数据字节 数据字 数据双字 数据位 数据字节 数据字 数据双字	DBX DBB DBW DBD DIX DIB DIW DID	0～65 535.7 0～65 535 0～65 534 0～65 532 0～65 535.7 0～65 535 0～65 534 0～65 532
本地数据（L）	本区域存储逻辑块（OB、FB 或 FC）中使用的临时数据，一般用作中间暂存器，因为这些数据实际存储在本地数据堆栈中，所以当逻辑块执行结束时，数据自然丢失	本地数据位 本地数据字节 本地数据字 本地数据双字	L LB LW LD	0～65 535.7 0～65 535 0～65 534 0～65 532

在使用存储区时注意以下几个方面。

① CPU 程序所能访问的存储区为系统存储区的全部、工作存储区的数据块（DB）、临时本地数据存储区、外设 I/O 存储区（P）等，外部输入寄存器（PI）和外部输出寄存器（PQ）存储区除了和 CPU 的型号有关外，还和具体的 PLC 应用系统的模块配置有关，其最大范围为 64 KB。

② CPU 可以通过输入（I）和输出（Q）过程映像存储区（映像表）访问 I/O 口。输入映像表 128B 是外部输入寄存器（PI）首 128B 的映像，是在 CPU 循环扫描中读取输入状态时装入的。输出映像表 128B 是外部输出存储区（PQ）首 128B 的映像。CPU 在写输出时，可以将数据直接输出到外部输出寄存器（PQ），也可以将数据传送到输出映像表，在 CPU 循环扫描更新输出状态时，将输出映像表的值传送到物理输出。

根据以上的分析可以看出，只有数字量模块既可以用 I/O 映像表也可以通过外部 I/O 存储器进行数据的输入、输出。而模拟量模块由于最小地址已超过了 I/O 映像表的最大值 128B，所以只能以字节、字或双字的形式通过外部 I/O 存储区直接存取。

③ 西门子 S7－300 PIB、PIW、PQW 与 IB、IW、QW 之间的区别如下。

外设最通常意义的讲就是指模拟量输入输出；普通的 I/O 最通常意义的讲是指 CPU 集成的 I/O 或者后面扩展数量字 I/O 模块的地址；两者的区别如下：

① I/O 区可以对“位”寻址，而 PI 和 PQ 区不能，PI 和 PQ 最小寻址单位是 BYTE，如 PIB256、PQW300。

② I/O 区，按西门子的定义应该叫“过程映像区”，本身这个过程映像区自己也是带有外设地址的。比如 IB0，它对应一个 PIB0，它们的不同之处在于每个 PLC 扫描周期的开始，系统会将 PIB0 里的数据刷新到 IB0 内，然后在接下来的一个扫描周期之内，IB0 的数据均保持不变（当然，如果调用 SFC 在程序内刷新输入端口则又当别论），而 PIB0 则是实时的，即完全物理意义上的端口。所以我们看到一些 S7－300 PLC 标出最大 I/O 寻址为 1024 位，例如，一块 CPU 带四个机架，每个机架装满八个模块，每个模块最大 32 点，那么 4×8×32＝1024 位＝128B…，超过这个以后就是外设通道了，就只能以 PI 或者 Q 来表示了，而且这就意味着不能直接对“位”寻址了。如果需要对位寻址，比如可以通过下列语句来实现：

```
L PIB256
T MB0
```

那么 MB0 里的位就对应了 PIB256 的 8 位数据。

5.6.2 CPU 中的寄存器

PLC 的 CPU 中包含有一些寄存器，CPU 使用这些寄存器便于执行逻辑运算、算术运算、装载和传输等操作。弄清这些寄存器的组成及功能对编写程序很有用。

（1）累加器（ACCUx）

累加器用于处理字节、字或双字的寄存器。S7－300 PLC 有两个 32 位累加器（ACCU1 和 ACCU2），S7－400 PLC 有 4 个 32 位累加器（ACCU1～ACCU4）。数据存储在累加器的低端（右对齐）。几乎所有的语句表的操作都是在累加器中进行的。

（2）地址寄存器

两个 32 位的地址寄存器 AR1 和 AR2 作为地址指针，用于寄存器间接寻址。

（3）数据块寄存器

32 位的 DB 和 DI 寄存器分别用于保存打开的共享数据块和背景数据块的编号。它们的高 16 位分别用于保存打开的共享数据块和背景数据块的编号，低 16 位用于保存打开的数据块的字节长度。

（4）状态字

状态字用于表示 CPU 执行指令时所具有的状态。一些指令是否执行或以何方式执行可能取决于状态字中的某些位；执行指令时也可能改变状态字中的某些位；也能在位逻辑指令或字逻辑指令中访问并检测它们。16 位的状态字只使用了其中的 9 位，9 ~ 15 位未使用，0 ~8 位 9 个位（bit）用于存储 CPU 执行指令的状态或结果及出现的错误。如图 5-7 所示。

15~9	8	7	6	5	4	3	2	1	0
未用	BR	CC 1	CC 0	OV	OS	OR	STA	RLO	$\overline{FC}$

图 5-7　状态字的结构

① 首次检测位（$\overline{FC}$）

状态字的位 0 称为首位检测位。若$\overline{FC}$位的状态为 0，则表明一个梯形逻辑网络的开始，或指令为逻辑串的第一条指令。CPU 对逻辑串第一条指令的检测（称为首位检测）产生的结果直接保存在状态字的 RLO 位中，经过首次检测存储在 RLO 中的 0 或 1 被称为首位检测结果。$\overline{FC}$位在逻辑串的开始时总是 0，在逻辑串指令执行过程中$\overline{FC}$位为 1，输出指令或与逻辑运算有关的转移指令（表示一个逻辑串结束的指令）$\overline{FC}$将清 0。

② 逻辑操作结果（RLO）

状态字的位 1 称为逻辑操作结果 RLO（Result of Logic Operation）。该位存储位逻辑指令或算术比较指令的结果。在逻辑串中，RLO 位的状态能够表示有关信号流的信息。RLO 的状态为 1，表示有信号流（通）；为 0，表示无信号流（断）。可用 RLO 触发跳转指令。

③ 状态位（STA）

状态位（第 2 位）用以保存被寻址位的值。状态位总是向扫描指令（A、AN、O…）或写指令（ = 、S、R）显示寻址位的状态（对于写指令，保存的寻址位状态是本条写指令执行后的该寻址位的状态）。

④ 或位（OR）

状态字的位 3 称为或位（OR）。在先逻辑“与”后逻辑“或”的逻辑串中，OR 位暂存逻辑“与”的操作结果，以便进行后面的逻辑“或”运算。其他指令将 OR 位清 0。

⑤ 溢出状态保持位（OS）

状态字的位 4 称为溢出状态保持位（或称为存储溢出位）。OV 被置 1 时 OS 也被置 1；OV 被清 0 时 OS 仍保持。所以它保存了 OV 位，可用于指明在先前的一些指令执行中是否产生过错误。只有下面的指令才能复位 OS 位：JOS（OS = 1 时跳转）；块调用指令和块结束指令。

⑥ 溢出位（OV）

状态字的位 5 称为溢出位。溢出位被置 1，表明一个算术运算或浮点数比较指令执行时

出现错误（错误：溢出、非法操作、不规范格式）。后面的算术运算或浮点数比较指令执行结果正常的话 OV 位就被清 0。

⑦ 条件码 1（CC1）和条件码 0（CC0）

状态字的位 7 和位 6 称为条件码 1 和条件码 0。这两位结合起来用于表示在累加器 1 中产生的算术运算或逻辑运算结果与 0 的大小关系；比较指令的执行结果或移位指令的移出位状态。表 5-4 和表 5-5。

表 5-4　算术运算后 CC1、CC0 状态代表的意义

CC1	CC0	算术运算无溢出	整数算术运算有溢出	浮点数算术运算有溢出
0	0	结果 =0	整数加时产生负范围溢出	平缓下溢
0	1	结果 <0	乘时负范围溢出；加、减、取负时正溢出	负范围溢出
1	0	结果 >0	乘、除时正溢出；加、减时负溢出	正范围溢出
1	1	——	在除时除数为 0	非法操作

表 5-5　比较、移位和循环移位、字逻辑指令后 CC1 和 CC0 状态代表的意义

CC1	CC0	比较指令	移位和循环移位指令	字逻辑指令
0	0	累加器 2 = 累加器 1	移出位为 0	结果为 0
0	1	累加器 2 < 累加器 1	——	——
1	0	累加器 2 > 累加器 1	——	结果不为 0
1	1	不规范（只用于浮点数比较）	移出位为 1	——

⑧ 二进制结果位（BR）

它将字处理程序与位处理联系起来，用于表示字操作结果是否正确（异常）。将 BR 位加入程序后，无论字操作结果如何，都不会造成二进制逻辑链中断。在 LAD 的方块指令中，BR 位与 ENO 有对应关系，用于表明方块指令是否被正确执行：如果执行出现了错误，BR 位为 0，ENO 也为 0；如果功能被正确执行，BR 位为 1，ENO 也为 1。

在用户用语句表编写的 FB（功能块）和 FC（功能）程序中，必须对 BR 位进行管理，当功能块正确运行后使 BR 位为 1，否则使其为 0。使用 STL 指令 SAVE 或 LAD 指令——（SAVE），可将 RLO 存入 BR 中，从而达到管理 BR 位的目的。当 FB 或 FC 执行无错误时，使 RLO 为 1 并存入 BR，否则，在 BR 中存入 0。

9 未用位。状态字的 9 ~ 15 位未使用。

5.7　指令结构与寻址方式

5.7.1　指令结构

（1）指令组成

用户程序是由若干条指令顺序排列而成的，而指令是程序的最小单元。下面分别介绍语句和梯形逻辑指令。

① 语句指令

一条指令由一个操作码和一个操作数组成，操作数由标识符和参数组成。操作码定义要执行的功能；操作数为执行该操作所需要的信息，例如："A　I 1.0"是一条位逻辑操作指令，其中："A"是操作码，它表示执行"与"操作；"I 1.0"是操作数，对输入继电器 I 1.0 进行的操作。有些语句指令不带操作数。它们操作的对象是唯一的。例如：NOT 默认是对逻辑操作结果（RLO）取反。

② 梯形逻辑指令

梯形逻辑指令用图形元素表示 PLC 要完成的操作。在梯形逻辑指令中，其操作码是用图素表示的，该图素形象表明 CPU 做什么，其操作数的表示方法与语句指令相同。如：

```
 Q4.0
—(S)—
```

该指令中：—(s) 可认为是操作码，表示一个二进制置位操作。Q4.0 是操作数，表示赋值的对象。梯形逻辑指令也可不带操作数。如：—｜NOT｜—是对逻辑操作结果取反的操作。

（2）操作数的表示法

① 标识符及表示参数

在一般情况下，指令的操作数在 PLC 的存储器中，此时操作数由操作数标识符和参数组成。操作数标识符由主标识符和辅助标识符组成。主标识符表示操作数所在的存储区，辅助标识符进一步说明操作数的位数长度。若没有辅助标识符；则指操作数的位数是一位。主标识符有 I（输入过程映像存储区）、Q（输出过程映像存储区）、M（位存储区）、PI（外部输入）、PQ（外部输出）、T（定时器）、C（计数器）、DB（数据块）、L（本地数据）。辅助标识符有 X（位）、B（字节）、W（字—2 字节）、D（双字—4 字节）。

当操作数长度是字或双字时，标识符后给出的标识参数是字或双字内的最低字节单元号。当使用宽度为字或双字的地址时，应保证没有生成任何重叠的字节分配，以免造成数据读/写错误。

② 操作数的表示法

在 STEP 7 中，操作数有两种表示方法：一是物理地址（绝对地址）表示法；二是符号地址表示法。

用物理地址表示操作数时，要明确指出操作数的所在存储区，该操作数的位数具体位置。例如：Q4.0。

STEP 7 允许用符号地址表示操作数，如 Q4.0 可用符号名 MOTOR_ON 替代表示，符号名必须先定义后使用，而且符号名必须是唯一的，不能重名。定义符号时，需要指明操作数所在的存储区，操作数的位数、具体位置及数据类型。

采用符号地址表示法可以使程序的可读性增强，并可以降低编程时由于笔误造成的程序错误。

5.7.2　寻址方式

一条完整的指令包含指令符和操作数（当然不包括那些单指令，如 NOT 等），操作数是指令的操作或运算对象。所谓寻址方式是指令得到操作数的方式，可以直接给出或间接

给出。

要描述一个地址，至少应该包含两个要素：存储的区域和这个区域中具体的位置。S7－300 PLC 有四种寻址方式：立即寻址、存储器直接寻址、存储器间接寻址和寄存器间接寻址。

（1） 立即寻址

立即寻址是对常数或常量的寻址方式，其特点是操作数直接表示在指令中，或以唯一形式隐含在指令中，下面各条指令操作数采用了立即寻址方式，其中“//”后面的内容为指令的注释部分，对指令没有任何影响。

```
SET                          //默认操作数为 RLO(逻辑输出),该指令对 RLO 置“1”操作
OW      W#16#A320            //将常量 W#16#A320 与累加器 1 进行“或”运算
L27                          //表示把常数 27 装入累加器 1 中
L       'ABCD'               //把 ASCII 码字符 ABCD 装入累加器 1
L       C#0100               //把 BCD 码常数 0100 装入累加器 1
```

（2） 直接寻址

存储器直接寻址，简称直接寻址，该寻址方式在指令中直接给出操作数的存储单元地址，地址通常是一个常数，不可以改变。存储单元地址可有符号地址（如 SB1、KM 等）或绝对地址（如 I0.0、Q4.1）。下面各条指令操作数均采用了直接寻址方式。例如：

```
A     I 0.0         //对输入位 I 0.0 进行“与”逻辑操作
S     L 20.0        //把本地数据位 L 20.0 置 1
=     M 115.4       //使存储区位 M 115.4 的内容等于 RLO 的内容
L     IB 10         //把输入字节 IB 10 的内容装入累加器 1
T     DBD 12        //把累加器 1 中的内容传送给数据双字 DBD 12 中
```

（3） 存储器间接寻址

存储器间接寻址指令中的存储内容为操作数所在存储单元的地址，该地址也称为地址指针，地址指针可以是字或双字，在指令中需写在方括号“[ ]”内。用于存储器间接寻址的地址给定格式为：地址标识符＋[16 位或 32 位地址指针]。指针所指示存储单元中的数值，就是地址的确切数值单元。存储器间接寻址具有两种格式：单字（16 位地址指针）和双字（32 位地址指针）。字或双字应在位存储区（M）、背景数据块（DIB）、数据块（DB）和局部数据（L）的区域之中。

单字：16 位地址指针用于定时器（T）、计数器（C）、数据块（DB）、功能（FC）、功能块（FB）的寻址，它被看作一个无符号整数（范围为 0～65535），表示定时器（T）、计数器（C）、数据块（DB）、功能（FC）、功能块（FB）的编号。16 位地址指针以 W 结尾，如 DBW。

双字：包含了地址标识符指明的在存储区域中确切的数值单元地址，通过指针指明数值或编号的单元。32 位双字指针的格式如图 5-8 所示。其中位 0～2（范围 0～7）表示的是被寻址位的位编号，位 3～18（范围 0～65535）表示的被寻址字节的字节编号，位 19～31 无定义。

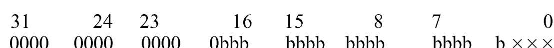


图 5-8　存储器间接寻址中双字指针的格式

寻址格式表示为：

```
A   I［MD 2］    //对由 MD 2 指出的输入位进行“与”逻辑操作。如：MD 2 的值为
                 //2#0000 0000 0000 0000 0000 0000 0101 0110 则是对 I 10.6 进行“与”操作。
```

再比如，指令 A M［LD 4］对位存储区 M 中的位做“与”操作，若地址指针 LD 4 的值为 2#0000 0000 0000 0000 0000 0000 0010 0011，根据 32 位指针格式，即对 M4.3 进行操作。

存储器间接寻址方式的优点是，当程序执行时，能改变操作数的存储器地址，这对程序中的循环尤为重要。

（4）寄存器间接寻址

在 S7 中有两个地址寄存器，它们是 AR1 和 AR2。通过地址寄存器，可以对各存储区的存储器内容实现寄存器间接寻址。地址寄存器的内容加上偏移量形成地址指针，该指针指向数值所在的存储单元。

地址寄存器存储的地址指针有两种格式：区内寄存器间接寻址和区域间寄存器间接寻址。其长度均为双字。指针格式如图 5-9 所示。说明如下。

位 31 =0 表明是区域内寄存器间接寻址；位 31 =1 表明是区域间寄存器间接寻址。

位 24、25 和 26（r r r）：被寻址地址的区域标识号。

位 3 ~18（bbbb bbbb bbbb bbbb）：被寻址位的字节编号（范围 0 ~65535）。

位 0 ~2（×××）：被寻址地址中的位编号（范围 0 至 7）。

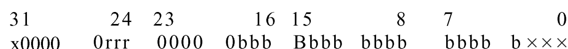


图 5-9　寄存器间接寻址中双字指针的格式

区域内寻址包括被寻址数据所在存储单元地址的字节编号和位编号，至于对哪个存储区寻址，则必须在指令中明确给出。这种格式适用于在确定的存储区内寻址，即区域内寄存器间接寻址，其表示为：区域标识符［ARx，地址偏移量］。例如：

```
L    P#5.0           //将 P#5.0 装入累加器 1
LAR1                 //将累加器 1 的内容传送至地址寄存器 1
A   I[AR1,P#2.2]     //对输入位 I7.2 进行“与”逻辑操作。P#5.0 + P#2.2 = P#7.2
 =Q［AR1,P#1.1］     //赋值给输出位 Q6.1。P#5.0 + P#1.1 = P#6.1
```

区域间寻址包含了数据所在存储区的说明位（存储区域标识位），可通过改变标识位实现跨区域寻址，其表示为：访问宽度［ARx，偏移量］。例如：

```
L    P#I8.7          //把指针值及存储区域标识装入累加器 1
LAR1                 //将累加器 1 的内容传送至地址寄存器 1
L    P#Q8.7          //把指针值及存储区域标识装入累加器 1
LAR2                 //将累加器 1 的内容传送至地址寄存器 2
A[AR1,P#0.0]         //检测 I8.7 的信号状态
 =［AR2,P#1.2］      //给输出位 Q10.1 赋值(注意:8.7 +1.2 =10.1,而不是 9.9)
```

5.8 S7－300 PLC 程序结构

为了便于阅读和理解，在编程中常常将程序分为若干部分。STEP 7 编程语言支持这种划分，并且提供了必要的功能。每个程序部分具有其技术和功能基础，称之为“块”。西门子公司 S7－300/400 系列 PLC 采用的就是“块式程序结构”，用“块”的形式来管理用户编写的程序及程序运行所需要的数据，组成完整的 PLC 应用程序系统。

块是程序中真正有用的部分，包括用户块和系统块，它们在功能、使用方法和结构上各不相同。

（1）用户程序使用的块

在 S7－300 PLC 中，PLC 的程序分为操作系统和用户程序。操作系统处理的是底层的系统级任务，它为 PLC 应用搭建了一个平台，提供了一套用户程序的调用机制，主要用于实现与特定的控制任务无关的功能，比如处理 PLC 的启动、刷新输入/输出过程映像表、调用用户程序、处理中断和错误、管理存储区和处理通信等。用户程序在这个搭建的平台上完成自己特定的自动化控制任务，而这个程序需要用户自己编写。通常用户程序主要完成暖启动和热启动的初始化工作、处理过程数据（数字信号、模拟信号）、对中断的响应、对异常和错误的处理等工作。

通常用户编写的程序和程序过程中所需的数据均被 STEP 7 放置在块中，使单个的程序标准化。这样做的好处是：块与块之间的调用使程序结构化、简单化、透明化，易于理解、修改、查错和调试。

用户程序是由块组成的，构成用户程序的逻辑块包括组织块 OB（Organization Block）、功能块 FB（Function Block）、功能 FC（Function）、系统功能块 SFB（System Function Block）、系统功能 SFC（System Function）。各种块的简单说明见表 5-6。

表 5-6 用户程序中的块

块	简要描述
组织块（OB）	操作系统与用户程序的接口，决定用户程序的结构
系统功能块（SFB）	集成在 CPU 模块中，通过 SFB 调用一些重要的系统功能，有存储区
系统功能（SFC）	集成在 CPU 模块中，通过 SFC 调用一些重要的系统功能，无存储区
功能块（FB）	用户编写的包含经常使用的功能的子程序，有存储区
功能（FC）	用户编写的包含经常使用的功能的子程序，无存储区
背景数据块（DI）	调用 FB 和 SFB 时用于传递参数的数据块，在编译过程中自动生成数据
共享数据块（DB）	存储用户数据的数据区域，供所有的块共享

组织块 OB 是操作系统和用户程序之间的界面，用于控制扫描循环和中断程序的执行、PLC 的启动和错误处理等，其他的组织块有启动组织块和中断组织块。档次较高的 CPU 可以使用的同类型组织块也就越多。操作系统只调用组织块，其他的程序块需要通过用户程序中的指令调用，操作系统才会加以处理（扫描）。其中最主要的组织块是 OB1，它是用户程序的主程序，也是操作系统自动做循环扫描的唯一的一个块。CPU 的操作系统完成启动过程后就循环执行 OB1，此时就可以在 OB1 中调用其他逻辑块。如果出现中断事件，当前正

在执行的组织块只有在当前指令执行完以后才会被停止，操作系统才会分配给该事件一个组织块。这个中断事件（组织块）执行完后，被中断的块将会从断点处继续执行。

功能 FC 和功能块 FB 是用户程序中主要的逻辑操作块，二者皆是用户编写的子程序，只是功能（FC）不需要背景数据块，无存储区，调用结束数据消失；而功能块始终存在如影随形的背景数据块，有存储区，主要的控制、运算、操作等均由它们来完成。组织块负责安排 FC 和 FB 的调用条件和调用顺序。对于功能块，操作系统为参数及静态变量分配的存储空间是背景数据块。操作系统在 L 堆栈中给 FB 的临时变量分配空间；对于功能，操作系统同样也在 L 堆栈中给 FC 的临时变量分配空间，只是 FC 不能使用静态变量。输入、输出、I/O 参数以指向实参的指针形式存储在操作系统为参数传递而保留的额外空间中。

系统功能块 SFB 和系统功能 SFC 是集成在 S7 CPU 的操作系统中，预先编好程序的逻辑块，它们不占用程序空间。其中系统功能块通过 SFB 调用系统功能，有专用的存储区（即背景数据块），而系统功能通过 SFC 调用系统功能，没有专用的存储区。用户不能打开它们，也不能修改它们内部的程序。通常用于完成一些通用的功能，如读写实时时钟、设置参数、数据通信等。在 S7 -300CPU 中通常会固化有一部分 SFB 和 SFC，用户在编程时可以调用。通常 SFC 和 SFB 提供一些系统级的功能调用，比如 SFC39“DIS_IRT”用于禁止中断和异步错误处理，可以禁止所有的中断、有选择地禁止某些范围的中断和某个中断；SFC40“EN_IRT”用于激活新的中断和异步错误处理，可以允许所有的中断和有选择地允许某些中断等。在调用 SFB 时，需要用户指定背景功能块（CPU 中不包含其背景数据块），并确定将背景数据块下载到 PLC 中。

数据块 DB 只用于存储用户数据。在数据块中只有数据和变量，没有程序。但数据块占用程序容量。数据块中没有 STEP 7 的指令，STEP 7 按数据生成的顺序自动地为数据块中的变量分配地址。数据块可以分为全局（共享）数据块和背景（伴随）数据块两种。

应首先生成功能块，然后再生成它的背景数据块。在生成背景数据块时应指明它的类型为背景数据块（Instance）和它的功能块的编号。

（2）块的调用

在系统上电启动之后，首先开始运行初始化程序 OB100，之后进入可编程的工作周期，OB1 是循环扫描的主程序块，它的优先级最低。各种块的关系如图 5-10 所示。组织块 OB 可以调用 SFB、SFC、FB 和 FC；而 FB 和 FC 也可以调用另外的 FB 和 FC，称为嵌套。FB 和 SFB 使用时需要配有相应的背景数据块（IDB）。

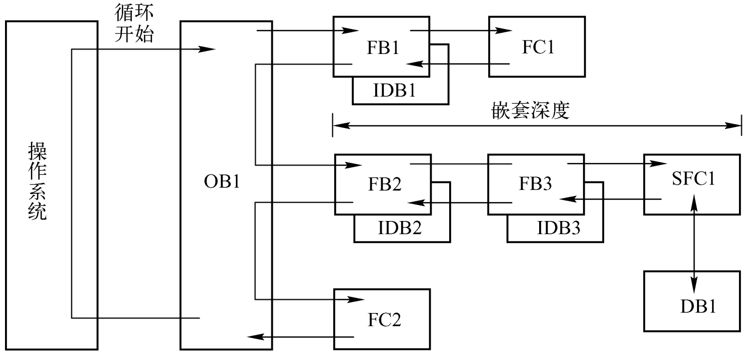


图 5-10　块的调用关系

（3）用户程序使用的堆栈

堆栈是 CPU 中的一块特殊的存储区，堆栈采用“先入后出”的规则存入和取出数据。最上面的存储单元称为栈顶。要保存的数据从栈顶“压入”堆栈时，堆栈中原有的数据依次向下移动一层，最下面的存储单元的数据丢失。在取出栈顶的数据后，堆栈中所有的数据依次向上移动一层。堆栈的这种“先入后出”的存取顺序，刚好满足块调用时（包括中断处理的块调用）存储和取出数据的要求，因此，堆栈在计算机的程序设计中得到了广泛的应用。

① 局部数据堆栈（L 堆栈）

各逻辑块都有它的局部变量（L）存储区，局部变量在逻辑块的变量声明表中生成，只在它被创建的块中有效。局域数据堆栈用于存储块的局域数据区的临时变量、组织块的启动信息、块传递参数的信息和梯形图程序的中间结果。每个组织块用 20B 的临时局部数据来存储它的启动信息。局部数据可以按位、字节、字和双字来存取，例如 L0.0、LB9、LW4 和 LD52。

CPU 分配给当前正在处理块的临时局部数据的存储容量是有限的，这一存储区的大小和 CPU 的型号有关。CPU 给每一个优先级分配了局部数据区，这样可以保证不同优先级的 OB 都有它们可以使用的局部数据空间。S7－300CPU 每一优先级的局部数据区的大小都是固定的。

② 块堆栈（B 堆栈）

如果一个块的处理因为调用另外一个块，或者被更高优先级的 OB 块中止，CPU 将在块堆栈中存储以下消息。

a. 被中断的块的类型（OB、FB、FC、SFB、SFC）、编号和返回地址。

b. 从 DB 和 DI 寄存器中获得的块被中断时打开的共享数据块和背景数据块的编号。

c. 局部数据堆栈的指针。

利用这些数据，可以在中断它的任务处理完后回复被中断的块的处理。在多重调用时，堆栈可以保存参与嵌套调用的几个块的信息。图 5-11 给出了堆栈中数据的动态变化的情况。图中 OB1 调用功能 FC2，FC2 的执行被电源故障组织块 OB1 中断。

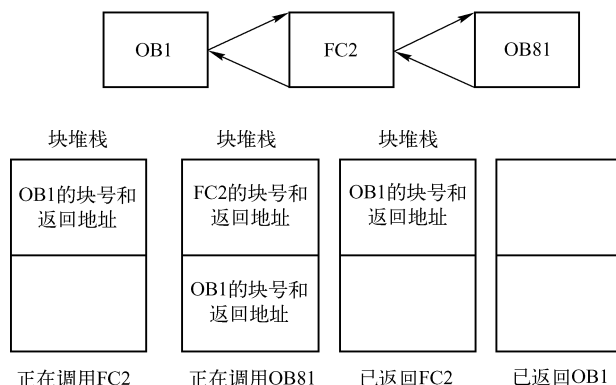


图 5-11　块堆栈中数据的动态变化

CPU 处于 STOP 模式时，可以在 CPU 的模块信息对话框中，查看堆栈保存的进入 STOP 模式时没有处理完的块，在块堆栈中，信息按照它们被处理的顺序存储。

③ 中断堆栈（I 堆栈）

如果程序的执行被优先级更高的 OB 中断，操作系统将保存下述寄存器的内容：当前的累加器和地址寄存器的内容、数据块寄存器 DB 和 DI 的内容、局部数据的指针、状态字、MCR（主控继电器）寄存器和块堆栈的指针。新的 OB 执行完后，操作系统读取中断堆栈中的信息，从被中断的地方开始继续执行程序。

CPU 在 STOP 或 HOLD 模式时，可以用 STEP 在 CPU 的模块信息中查看中断堆栈保存的数据，由此找出使 CPU 进入 STOP 模式的原因。

习题

5-1　说明 PLC 编程语言的种类及各自的特点，哪两种语言最常用？

5-2　梯形图的编程原则有哪些？

5-3　STEP 7 中的数据类型有哪三大类？基本数据类型包括哪些种类？

5-5　什么是状态字寄存器？简述状态字各位的含义。

5-5　STEP 7 中有哪些寻址方式？分述各方式的含义。

5-6　论述 S7－300 PLC 的程序结构。

第 6 章　S7－300 PLC 指令系统

S7－300 PLC 具有丰富的指令系统，其中包括逻辑指令和功能指令两大类。逻辑指令包括位逻辑指令、定时器指令、计数器指令、字逻辑指令。功能指令主要包括数据处理与算术运算指令、程序执行控制指令以及其他功能指令。

6.1　位逻辑指令

6.1.1　位逻辑指令简介

位逻辑指令处理的对象是“1”和“0”数字信号，这两个数字组成了二进制计数系统中的“位”，可代表输入触点的“闭合”和“断开”，或输出线圈的“通电”和“断电”。位逻辑指令主要包括位逻辑运算指令、位操作指令和位测试指令。它们完成逻辑运算操作，将运算结果保存在状态字的 RLO 中，逻辑操作结果（RLO）用以赋值、置位、复位布尔操作数，也控制定时器和计数器的运行。

在 STEP 中位逻辑指令有以下几种，图 6-1 所示为 STEP 7 中逻辑指令梯形图形式。

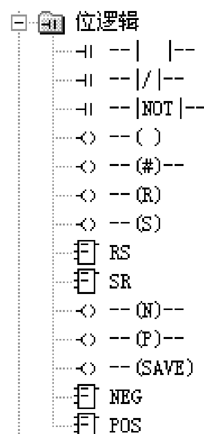


图 6-1　STEP 7 中逻辑指令梯形图形式

（1）—|　|—常开触点

当存储在指定 < 地址 > 的位置为“1”时，（常开触点）处于闭合状态。触点闭合时，梯形图轨道能流过触点，逻辑运算结果（RLO） = “1”。否则，如果指定 < 地址 > 的信号状态为“0”，触点将处于断开状态。触点断开时，能流不流过触点，逻辑运算结果（RLO） = “0”。

串联使用时，通过 AND 逻辑将—|　|— 与 RLO 位进行链接。并联使用时，通过 OR 逻辑将其与 RLO 位进行链接。

常开触点所使用的操作数是：I、Q、M、L、D、T、C。

（2）—| / |—常闭触点

当存储在指定 < 地址 > 的位置为“0”时，（常闭触点）处于闭合状态。触点闭合时，梯形图轨道能流过触点，逻辑运算结果（RLO） = “1”。否则，如果指定 < 地址 > 的信号状态为“1”，将断开触点。触点断开时，能流不流过触点，逻辑运算结果（RLO） = “0”。

串联使用时，通过 AND 逻辑将 —| / |— 与 RLO 位进行链接。并联使用时，通过 OR 逻辑将其与 RLO 位进行链接。

常闭触点所使用的操作数是：I、Q、M、L、D、T、C。

（3）—|NOT|— 能流取反

该指令用于将它左侧电路的逻辑运算结果（RLO）取反，该运算结果若为 1 则变为 0，

若为 0 则变为 1。

（4）—(　)输出线圈

输出线圈的工作方式与继电器逻辑图中线圈的工作方式类似。如果有能流通过线圈（RLO = 1），将置位 <地址> 位置的位为“1”。如果没有能流通过线圈（RLO = 0），将置位 <地址> 位置的位为“0”。只能将输出线圈置于梯级的右端。可以有多个（最多 16 个）输出单元，使用—| NOT |—（能流取反）单元可以创建取反输出。

输出线圈所使用的操作数是：I、Q、M、L、D。

（5）—(#)—中间输出

中间标有“#”的中间线圈是一种中间分配单元，它将 RLO 位状态（能流状态）保存到指定 <地址>。中间输出单元保存前面分支单元的逻辑结果。以串联方式与其他触点连接时，可以像插入触点那样插入—(#)—，并不影响能流的流到。不能将 —(#)— 单元连接到电源轨道、直接连接在分支连接的后面或连接在分支的尾部。使用—| NOT |—（能流取反）单元可以创建取反 —(#)—。

中间输出所使用的操作数是：I、Q、M、L、D。

（6）—(R)复位指令

只有在前面指令的 RLO 为“1”（能流通过线圈）时，才会执行—(R)（复位线圈）。如果能流通过线圈（RLO 为“1”），将把单元的指定 <地址> 复位为“0”。RLO 为“0”没有能流通过线圈）将不起作用，单元指定地址的状态将保持不变。<地址> 也可以是置复位为“0”的定时器（T 编号）或置复位为“0”的计数器（C 编号）。

复位指令所使用的操作数是：I、Q、M、L、D、T、C。

（7）—(S)置位指令

只有在前面指令的 RLO 为“1”（能流通过线圈）时，才会执行—(S)（置位线圈）。如果 RLO 为“1”，将把单元的指定 <地址> 置位为“1”，如果此时 RLO = 0，它仍然保持 1 状态。RLO = 0 将不起作用，单元的指定地址的当前状态将保持不变。

置位指令所使用的操作数是：I、Q、M、L、D。

（8）RS 置位优先指令

如果 R 输入端的信号状态为“1”，S 输入端的信号状态为“0”，则复位 RS（置位优先型 RS 双稳态触发器）。否则，如果 R 输入端的信号状态为“0”，S 输入端的信号状态为“1”，则置位触发器。如果两个输入端的 RLO 状态均为“1”，则指令的执行顺序是最重要的。RS 触发器先在指定 <地址> 执行复位指令，然后执行置位指令，以使该地址在执行余下的程序扫描过程中保持置位状态。

只有在 RLO 为“1”时，才会执行 S（置位）和 R（复位）指令。这些指令不受 RLO“0”的影响，指令中指定的地址保持不变。

复位优先指令所使用的操作数是：S 是 I、Q、M、L、D；R 是 I、Q、M、L、D；Q 是 I、Q、M、L、D。

（9）SR 复位优先指令

如果 S 输入端的信号状态为“1”，R 输入端的信号状态为“0”，则置位 SR（复位优先型 SR 双稳态触发器）。否则，如果 S 输入端的信号状态为“0”，R 输入端的信号状态为“1”，则复位触发器。如果两个输入端的 RLO 状态均为“1”，则指令的执行顺序是最重要

的。SR 触发器先在指定 < 地址 > 执行置位指令，然后执行复位指令，以使该地址在执行余下的程序扫描过程中保持复位状态。

只有在 RLO 为“1”时，才会执行 S（置位）和 R（复位）指令。这些指令不受 RLO “0”的影响，指令中指定的地址保持不变。

复位优先指令所使用的操作数同置位优先指令。

（10）—(N)— RLO 负跳沿检测

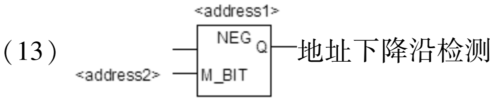
该指令检测地址中“1”到“0”的信号变化，并在指令后将其显示为 RLO = “1”。将 RLO 中的当前信号状态与地址的信号状态（边沿存储位）进行比较。如果在执行指令前地址的信号状态为“1”，RLO 为“0”，则在执行指令后 RLO 将是“1”（脉冲），在所有其他情况下将是“0”。指令执行前的 RLO 状态存储在地址中。

（11）—(P)— RLO 正跳沿检测

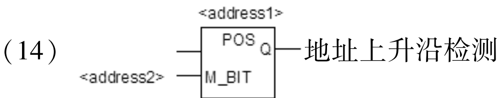
该指令检测地址中“0”到“1”的信号变化，并在指令后将其显示为 RLO = “1”。将 RLO 中的当前信号状态与地址的信号状态（边沿存储位）进行比较。如果在执行指令前地址的信号状态为“0”，RLO 为“1”，则在执行指令后 RLO 将是“1”（脉冲），在所有其他情况下将是“0”。指令执行前的 RLO 状态存储在地址中。

（12）—(SAVE)：将 RLO 状态保存到 BR

该指令将 RLO 保存到状态字的 BR 位。建议用户不要在使用 SAVE 后在同一块或从属块中校验 BR 位，因为这期间执行的指令中有许多会对 BR 位进行修改。建议用户在退出块前使用 SAVE 指令，因为 ENO 输出（= BR 位）届时已设置为 RLO 位的值，所以可以检查块中是否有错误。

（13）


该指令比较 < address1 > 的信号状态与前一次扫描的信号状态（存储在 < address2 > 中）。如果当前 RLO 状态为“1”且其前一状态为“0”（检测到上升沿），执行此指令后 RLO 位将在一个扫描周期内是“1”。

（14）


该指令比较 < address1 > 的信号状态与前一次扫描的信号状态（存储在 < address2 > 中）。如果当前 RLO 状态为“0”且其前一状态为“1”（检测到上升沿），执行此指令后 RLO 位将在一个扫描周期内是“1”。

下面分类进行详细的讲解和使用举例。

6.1.2 位逻辑指令使用方法

位逻辑指令是“与”（AND）、“或”（OR）、“异或”（XOR）指令及其组合。它对“0”或“1”这些布尔操作数扫描，经逻辑运算后将逻辑操作结果送入状态字的 RLO 位。

A（And，与）指令表示串联的常开触点。

O（Or，或）指令表示并联的常开触点。

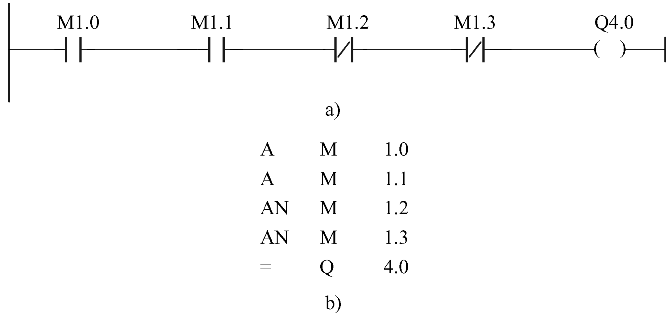
AN（And Not，与非）指令表示串联的常闭触点。

ON（Or Not）指令表示并联的常闭触点。

输出指令“ = ”将 RLO 写入地址位，与线圈相对应。

（1）“与”和“与非”（A，AN）指令

逻辑“与”在梯形图里是用串联的触点回路表示的，如果串联回路里的所有触点皆闭合，该回路就通“电”了。如图 6-2 所示。



a)

```
A    M    1.0
A    M    1.1
AN   M    1.2
AN   M    1.3
=    Q    4.0
```

b)

图 6-2　电路的串联

a）梯形图　b）语句表

（2）“或”和“或非”（O，ON）指令

逻辑“或”在梯形图里是用并联的触点回路表示的，被扫描的操作数标在触点上方。

（3）嵌套指令

嵌套指令只有 STL 指令，用于电路块串、并联的编程。它有“与嵌套”和“或嵌套”两种指令。

①“与嵌套”指令用于电路块串联的编程。其指令格式如下：

A（ ——与嵌套开始指令

　）——与嵌套结束指令

②“或嵌套”指令用于电路块并联的编程。其指令格式如下：

O（ ——或嵌套开始指令

　）——或嵌套结束指令

嵌套指令的用法如图 6-3 所示。

说明：先与后或（即电路元件先串后并），可不用嵌套指令中的括号。

（4）“异或”和“异或非（同或）”指令

①“异或”指令

异或指令只有 STL 指令，专用于异或门逻辑电路的编程。图 6-4 是“异或”逻辑梯形图和对应的语句表。当 I0.0 和 I0.1 不同时动作时，输出线圈 Q4.0 状态为 1，反之为 0。

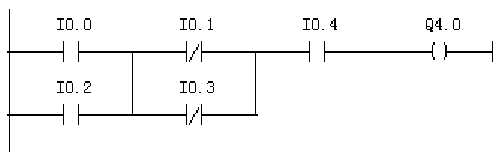
在语句表中，使用了“异或”和“同非”指令，分别用助记符“X”和“XN”来标识。它类似“或”和“或非”指令，用于扫描并联回路能否“通电”。

②“同或”指令

同或指令只有 STL 指令，专用于同或门逻辑电路的编程。图 6-5 说明了同或指令的用法。当 I0.0 和 I0.1 同时动作时，输出线圈 Q4.0 状态为 1，反之为 0。

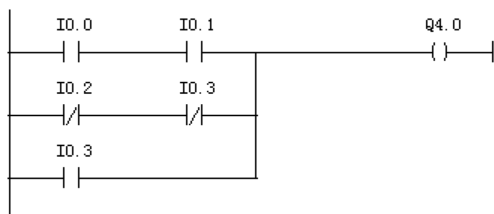
程序段 1：标题：

注释：



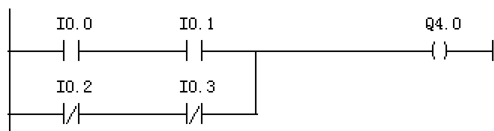
程序段 2：标题：

注释：



程序段 3：标题：

注释：



程序段 1：标题：

注释：

```
A(
O    I    0.0
O    I    0.2
)
A(
ON   I    0.1
ON   I    0.3
)
A    I    0.4
=    Q    4.0
```

程序段 2：标题：

注释：

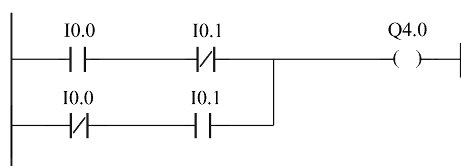
```
O(
A    I    0.0
A    I    0.1
)
O(
AN   I    0.2
AN   I    0.3
)
O    I    0.3
=    Q    4.0
```

程序段 3：标题：

注释：

```
A    I    0.0
A    I    0.1
O
AN   I    0.2
AN   I    0.3
=    Q    4.0
```

图 6-3　嵌套指令用法示例



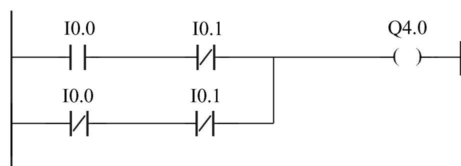
a)

```
       X    I    0.0
       X    I    0.1
       =    Q    4.0
或者
       XN   I    0.0
       XN   I    0.1
       =    Q    4.0
```

b)

图 6-4　“异或”指令的用法

a）梯形图　b）语句表



a)

```
       X    I    0.0
       XN   I    0.1
       =    Q    4.0
或者
       XN   I    0.0
       X    I    0.1
       =    Q    4.0
```

b)

图 6-5　“同或”指令的用法

a）梯形图　b）语句表

6.1.3 位操作指令

（1）输出指令

逻辑串输出指令又称为赋值操作指令，该操作把状态字中 RLO 的值赋给指定的操作数（位地址）。表 6-1 列出了操作数的数据类型和所在的存储区。

表 6-1 输出指令的功能说明

LAD 指令	STL 指令	功能	类型	存储区
<位地址>—( )	= <位地址>	逻辑串赋值输出	BOOL	Q、M、D、L
<位地址>—(#)—	—	中间结果赋值输出	BOOL	Q、M、D、L

说明：

① 中间输出指令被安置在逻辑串中间，用于将其前的位逻辑操作结果（此处的 RLO 值）保存到指定位地址（有人称它为“连接器”或“中间赋值元件”）。

②“连接器”和其他元件串联时，中间输出指令同触点一样，可插入逻辑串中间。

③“连接器”不能直接与左母线（相当于电路的电源母线）相连，也不能放在逻辑串的结尾或分支结尾处。

④ 可以用取反指令“—|NOT|—”对“连接器”进行取反操作。

⑤ 使用中间输出指令可以使复杂逻辑块程序简化成若干个简单逻辑块程序。

⑥ 一个 RLO 可被用于驱动几个输出元件。在 LAD 中，输出线圈是上下依次排列的。在 STL 中，与输出信号有关的指令被一个接一个地连续编程，这些输出具有相同的优先级。图 6-6 是多重输出梯形图和与之对应的语句表。

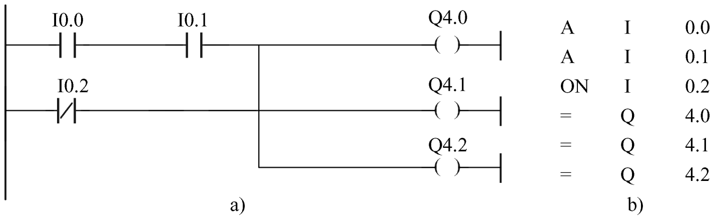


图 6-6 多重输出示例

a）梯形图 b）语句表

（2）置位/复位指令

置位/复位指令根据 RLO 的值，来决定被寻址位的信号状态是否需要改变。若 RLO 的值为 1，被寻址位的信号状态被置 1 或清 0；若 RLO 是 0，则被寻址位的信号保持原状态不变。置位/复位指令说明见表 6-2。

表 6-2 置位/复位指令

LAD 指令	STL 指令	功 能	类 型	存 储 区
<位地址> —(R)	R <位地址>	复位输出	BOOL TIMER COUNTER	Q、M、D、L T、C
<位地址> —(S)	S <位地址>	置位输出	BOOL	Q、M、D、L

【例 6-1】图 6-7 表示一个能够电气启动的传送带。在传送带的起点有两个按钮开关：用于 START 的 S1（I0.1）和用于 STOP 的 S2（I0.2）。在传送带的尾部也有两个按钮开关：S3（I0.3）用于 START，S4（I0.4）用于 STOP。可以从任一端启动或停止传送带。另外，当传送带上的物件到达末端时，传感器 S5（I0.5）使传送带停机。

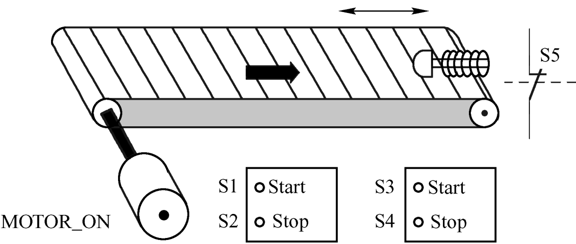


图 6-7　传送带控制示意图

控制程序均在 OB1 组织块内完成，相应的程序如图 6-8 所示，其中 Q4.0 用于传送带驱动电动机的控制。

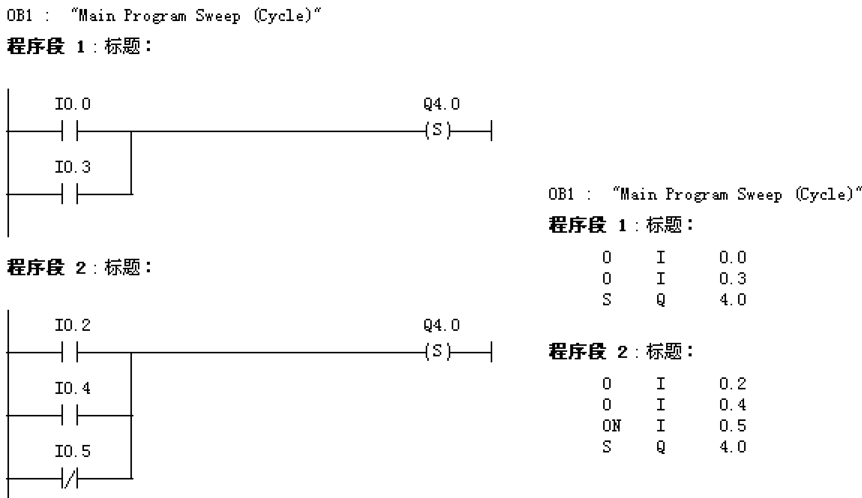


图 6-8　置位/复位指令使用举例

（3）RS 触发器

RS 触发器梯形图方块指令见表 6-3。方块中标有一个置位输入（S）端，一个复位输入（R）端，输出端标为 Q。触发器可以用在逻辑串最右端，结束一个逻辑串，也可用在逻辑串中，影响右端的逻辑操作结果。

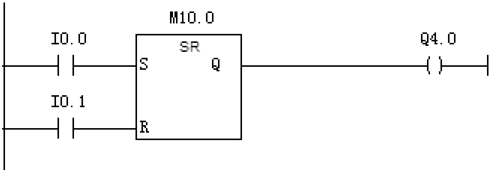
表 6-3　RS 触发器

复位优先型 SR	置位优先 RS	参　　数	数 据 类 型	存　储　区
〈位地址〉 SR S　Q R	〈位地址〉 RS S　Q R	< 位地址 > 需要置位、复位的位	BOOL	Q、M、D、L
		S 允许置位输入		
		R 允许复位输入		
		Q < 地址 > 的状态		

RS 和 SR 触发器指令的基本功能与置位指令 S 和复位指令 R 的功能相同，所以编程时，RS 和 SR 指令完全可以被置位和复位指令代替。

图 6-9 给出了使用例子。

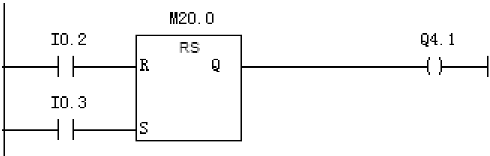
程序段 1：标题：



程序段 1：标题：

```
A    I    0.0
S    M   10.0
A    I    0.1
R    M   10.0
A    M   10.0
=    Q    4.0
```

程序段 2：标题：



程序段 2：标题：

```
A    I    0.2
R    M   20.0
A    I    0.3
S    M   20.0
A    M   20.0
=    Q    4.1
```

图 6-9　触发器梯形图和语句表示例

（4）对 RLO 的直接操作指令

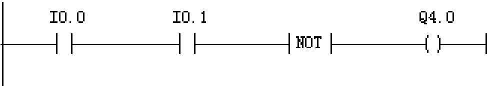
这一类指令直接对逻辑操作结果 RLO 进行操作，改变状态字中 RLO 位的状态。见表 6-4。

表 6-4　对 RLO 的直接操作指令

LAD 指令	STL 指令	功　能	说　明
—|NOT|—	NOT	取反 RLO	在逻辑串中，对当前的 RLO 取反，将当前 RLO 值由“0”变为“1”，或由“1”变为“0”
—	SET	置位 RLO	把 RLO 无条件置 1 并结束逻辑串；使 STA 置 1，OR、FC 清 0
—	CLR	复位 RLO	把 RLO 无条件清 0 并结束逻辑串；使 STA、OR、FC 清 0
—(SAVE)	SAVE	保存 RLO	把 RLO 存入状态字的 BR 位，该指令不影响其他状态位。如果没有存储 BR 位信号，编写的函数或函数块使用 LAD 程序直接调用时，ENO 不能直接输出，函数显示为虚线

图 6-10 给出了使用例子。

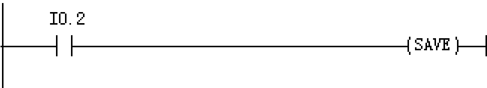
程序段 1：标题：



程序段 1：标题：

```
A    I    0.0
A    I    0.1
NOT
=    Q    4.0
```

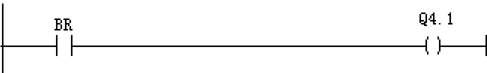
程序段 2：标题：



程序段 2：标题：

```
A    I    0.2
SAVE
```

程序段 3：标题：



程序段 3：标题：

```
A    BR
=    Q    4.1
```

图 6-10　对 RLO 的直接操作示例

6.1.4 位测试指令

当信号状态变化时就产生跳变沿。当从 0 变到 1 时，产生一个上升沿（或正跳沿）；若从 1 变到 0，则产生一个下降沿（或负跳沿）。如图 6-11 所示。

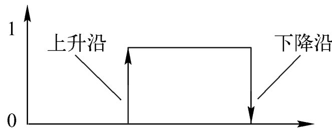


图 6-11　边沿信号

STEP 7 中有两类跳变沿检测指令，一种是对 RLO 的跳变沿检测的指令，另一种是对触点跳变沿直接检测的梯形图方块指令。见表 6-5。

表 6-5　位测试指令

<table>
<tr><td>LAD 指令</td><td colspan="2">STL 指令</td><td>功　能</td><td>操 作 数</td><td>数 据 类 型</td><td>存 储 区</td></tr>
<tr><td><位地址>
—(P)—</td><td colspan="2">FP <位地址></td><td>RLO 正跳沿检测</td><td><位地址></td><td>BOOL</td><td>I、Q、M、D、L</td></tr>
<tr><td><位地址>
—(N)—</td><td colspan="2">FN <位地址></td><td>RLO 负跳沿检测</td><td><位地址></td><td>BOOL</td><td>I、Q、M、D、L</td></tr>
<tr><td colspan="2">触点负跳沿检测</td><td colspan="2">触点正跳沿检测</td><td>参数</td><td>数据类型</td><td>存储区</td></tr>
<tr><td colspan="2" rowspan="3"><位地址1>
MEG
Q
<位地址2> M_BIT</td><td colspan="2" rowspan="3"><位地址1>
POS
Q
<位地址2> M_BIT</td><td><位地址 1>
被检测的位（触点）</td><td>BOOL</td><td>I、Q、M、D、L</td></tr>
<tr><td>M_BIT
存储被检测位上一个扫描周期的状态</td><td>BOOL</td><td>Q、M、D</td></tr>
<tr><td>Q 单稳输出</td><td>BOOL</td><td>I、Q、M、D、L</td></tr>
</table>

边沿检测的原理是：在每个扫描周期中把信号状态和它在前一个扫描周期的状态进行比较，若不同则表明有一个跳变沿。因此，前一个周期里的信号状态必须被存储，以便能和新的信号状态相比较。

注意：边沿检测常用于只扫描一次的情况，比如，在程序开始给一个变量赋了初值，如果不加边沿检测指令，由于 PLC 顺序循环扫描的特点，变量将永远是初始值，不发生任何变化。

【例 6-2】行车控制，用 STEP 7 编写程序，要求如下。

用按钮对行车的大车进行左移、右移控制，并用按钮对行车的小车进行上升、下降控制。

分析：根据控制要求可知，要实现行车控制，首先要知道，大车小车的启停条件，大车的启动条件为 I0.0 或 I0.1 按下，停止条件是 I0.2 按下或碰到限位开关（M200.0、M200.1）。小车与大车原理相似。

在 STEP 7 的 I/O 分配见表 6-6。

表 6-6　I/O 分配表

状　　态	符　　号	地　　址	数据类型
1	大车停止	I0. 2	BOOL
2	大车右移	I0. 0	BOOL
3	大车右移动作	Q5. 0	BOOL
4	大车左移	I0. 1	BOOL
5	大车左移动作	Q5. 1	BOOL
6	上限位	M200. 2	BOOL
7	下限位	M200. 3	BOOL
8	小车上升	I0. 4	BOOL
9	小车上升动作	Q5. 2	
10	小车停止	I0. 6	BOOL
11	小车下降	I0. 5	BOOL
12	小车下降动作	Q5. 3	BOOL
13	右限位	M200. 0	BOOL
14	左限位	M200. 1	BOOL
15			

梯形图程序如图 6-12 所示。

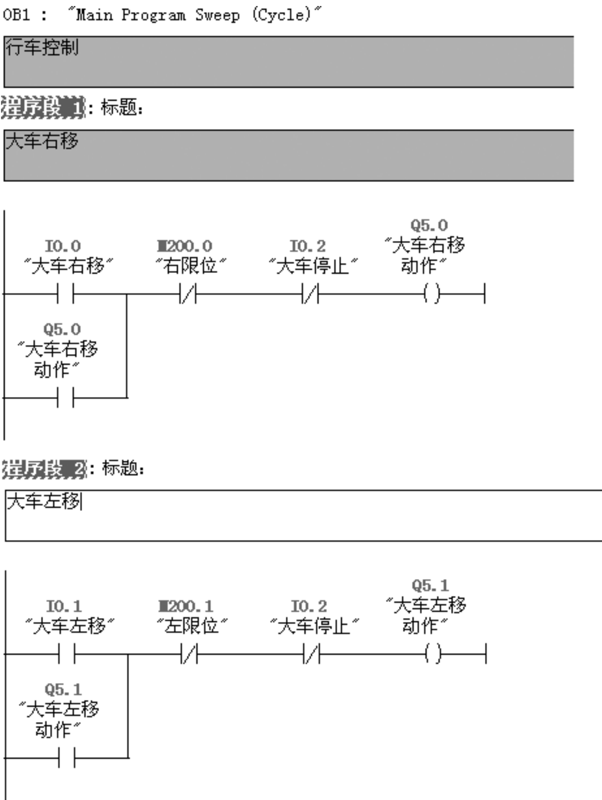
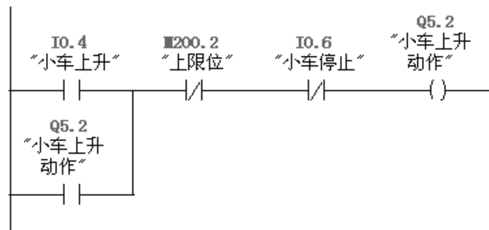


图 6-12　行车控制梯形图程序

程序段 3：标题：

小车上升



程序段 4：标题：

小车下降

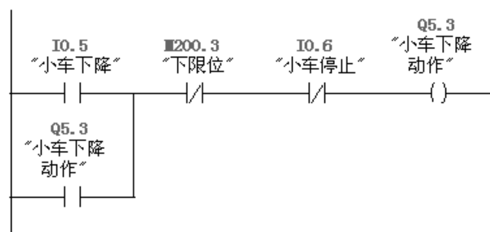
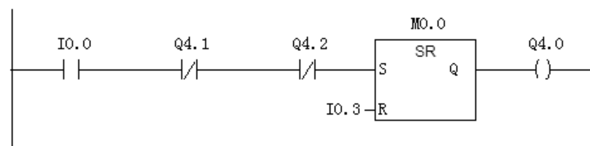


图 6-12　行车控制梯形图程序（续）

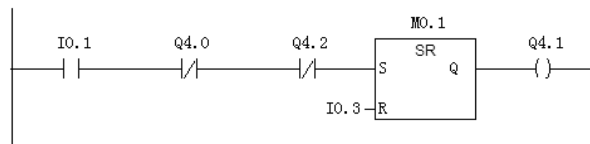
【**例 6-3**】抢答器有三个输入，分别为 I0.0、I0.1 和 I0.2，输出分别为 Q4.0、Q4.1 和 Q4.2，复位输入是 I0.3。要求：三人中任意抢答，谁先按按钮，谁的指示灯优先亮，且只能亮一盏灯，进行下一问题时主持人按复位按钮，抢答重新开始。实现其功能的梯形图如图 6-13 所示。

OB1 : "Main Program Sweep (Cycle)"

程序段 1：标题：



程序段 2：标题：



程序段 3：标题：

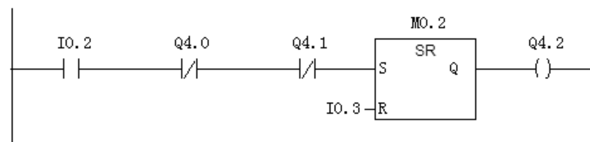


图 6-13　抢答器梯形图

也可以使用图6-14所示的梯形图来实现，程序的编写不拘一格，关键在于合理的组织，能实现控制功能即可。

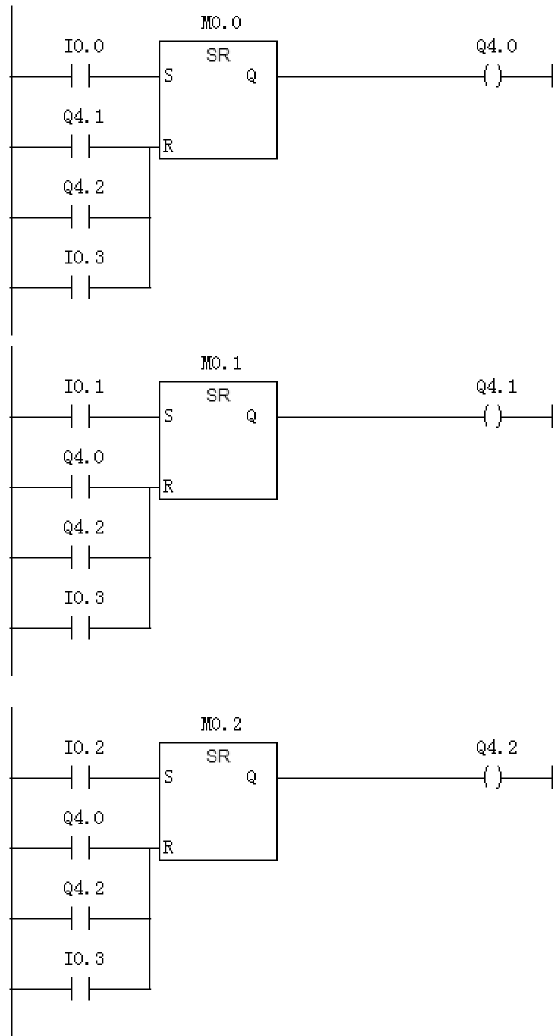


图6-14　抢答器梯形图

注意：SR指令上的存储位不能重复，否则程序出错。

【例6-4】设计一个乒乓电路，按动按钮I0.0，使灯泡亮，再按动按钮，灯泡灭。实现其功能的梯形图如图6-15所示。对于图6-15所示的梯形图，分析如下。

① 第一次按动按钮时，I0.0接通，在一个扫描周期中，则M0.1通；M0.2接通；Q4.0通，电灯亮。

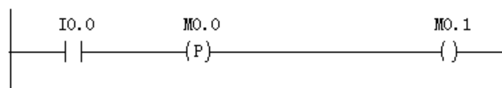
② 当一个扫描周期结束时，M0.1断开；在第二分支中，由于M0.2是通的，同时常闭触点M0.1是通的，M0.2保持接通状态，Q4.0通，电灯继续保持亮。

③ 当I0.0再一次按动时，M0.1接通；在Network2中第一分支和第二分支能流不通，M0.2断开，Q4.0不通，电灯灭，之后系统循环运行。

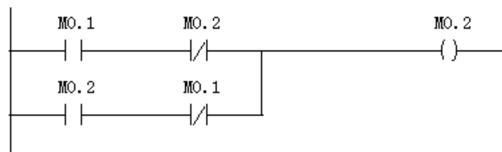
也可以使用图6-16所示的梯形图来实现。

OB1 : "Main Program Sweep (Cycle)"

程序段 1：标题：



程序段 2：标题：



程序段 3：标题：



图 6-15　闪烁灯控制梯形图

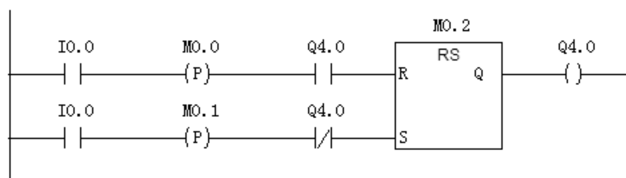


图 6-16　闪烁灯控制梯形图

6.2　定时器指令

6.2.1　定时器指令简介

定时器相当于继电器电路中的时间继电器，是 PLC 中的重要部件，它用于实现或监控时间序列。S5 是西门子 PLC 老产品的系列号，S5 定时器是西门子 S5 系列 PLC 的定时器。定时器是一种由位和字组成的复合单元，定时器的触点由位表示，其定时时间值存储在字存储器中。定时器的数目由 CPU 决定，S7－300 PLC 定时器的个数（128～2048）与 CPU 的型号有关。S7－300 PLC 中定时器包括以下几种，如图 6-17 所示分别为定时器的梯形图和线圈指令形式。

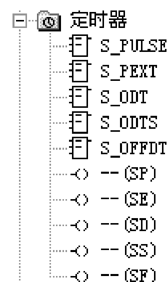


图 6-17　STEP 7 中定时器指令

（1）定时器指令分类

S7－300 PLC 提供的定时器有五种：脉冲定时器（SP）、扩展定时器（SE）、接通延时定时器（SD）、带保持的接通延时定时器（SS）和断电延时定时器（SF）。这五种定时器的梯形图、数据类型、参数、描述及存储区见表 6-7。

表 6-7　定时器指令说明

名　　称	梯　形　图	数据类型	参　　数	存　储　区	描　　述
1. S_PULSE； 脉冲 S5 定时器	S_PULSE S　Q TV　BI R　BCD	定时器	T 编号	T	定时器标识号，范围取决于 CPU
2. S_PEXT； 扩展脉冲 S5 定时器	S_PEXT S　Q TV　BI R　BCD	S	BOOL	I、Q、M、L、D	使能输入，起动输入端
		TV	S5TIME	I、Q、M、L、D	预设时间值
3. S_ODT； 接通延时 S5 定时器	S_ODT S　Q TV　BI R　BCD	R	BOOL	I、Q、M、L、D	B 复位输入端
		BI	WORD	I、Q、M、L、D	剩余时间值，整型格式
4. S_ODTS； 保持接通延时 S5 定时器	S_ODTS S　Q TV　BI R　BCD	BCD	WORD	I、Q、M、L、D	剩余时间值，BCD 格式
5. S_OFFDT； 断开延时 S5 定时器	S_OFFDT S　Q TV　BI R　BCD	Q	BOOL	I、Q、M、L、D	定时器的状态

（2）定时器字的表示方法

在 CPU 的存储器中留出了定时器区域，用于存储定时器的定时时间值。每个定时器为 2B，称为定时字。S7 中定时时间由时基和定时值两部分组成，定时时间等于时基与定时值的乘积。采用减计时，当定时器运行时，定时值不断减 1，直至减到 0，减到 0 表示定时时间到。定时时间到后会引起定时器触点的动作。

定时器的第 0 到第 11 位存储 BCD 码格式的定时值，三位 BCD 码表示的范围是 0 ~ 999，第 12，13 位存储二进制格式的时基。如图 6-18 所示。表 6-8 给出了时基的表示方法。

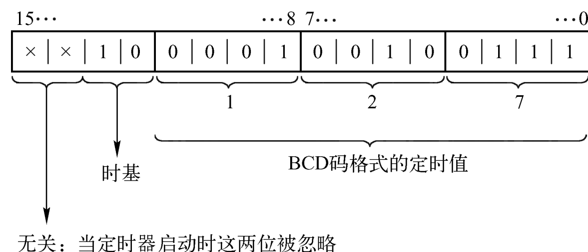


图 6-18　定时器字

表 6-8　时基的表示

时　　基	二进制时基	分　辨　率	定 时 范 围
10 ms	00	0. 01 s	10 ms 至 9 s_990 ms
100 ms	01	0. 1 s	100 ms 至 1 m_39 s_900 ms
1 s	10	1 s	1 s 至 16 m_39 s
10 s	11	10 s	10 s 至 2 h_46 m_30 s

从表 6-8 中可以看出：时基小定时分辨率高，但定时时间范围窄；时基大分辨率低，但定时范围宽。

当定时器启动时，累加器 1 低字的内容被当作定时时间装入定时字中。这一过程是由操作系统控制自动完成的，用户只需给累加器 1 装入不同的数值，即可设置需要的定时时间。定时值的设定可通过以下两种方法进行。

① 直接表示法。直接表示法仅在语句表指令（STL）中使用，其指令格式如下：

```
L   W#16#wxyz      //执行后,把 wxyz 存入累加器 1 低字(即低 16 位)中,其中 xyz 以 BCD 码形式
                     存入,w 以二进制码形式存入
```

其中：w = 时基，取值 0、1、2、3，分别表示时基为 10 ms、100 ms、1 s、10 s；xyz = 定时值，是 BCD 码格式的时间值，取值范围：1 ~ 999。

例如：
```
A     I0. 0          //允许 T4 启动的输入控制信号
L     W#16#2127      //把 2127 存入累加器 1 低字中
SP    T4             //启动 T4，且累加器 1 存储的 2127 自动装入定时器字中
```

② S5 时间表示法。S5 时间表示法在 STL、LAD 以及梯形图方块指令中都能用。其指令格式如下：

```
L     S5T# aH_bbM_ccS_dddMS   //执行后,把定时值 aHbbMccSdddMS 以二进制数的形式存入累
                                加器 1 低字(即低 16 位)中
```

其中：a = 小时，bb = 分钟，cc = 秒，ddd = 毫秒；设定范围为 10MS 到 2H_46M_30S；此时，时基是自动选择的，原则是：根据定时时间选择能满足定时范围要求的最小时基。在梯形图中必须使用“S5T#”格式的时间预置值 S5T# aH_bbM_ccS_dddMS，在输入时可以不输入下划线。

说明：该指令执行是把定时值以二进制数的形式装入累加器 1 中，当执行后面的定时器指令时，累加器 1 存储的定时值会以二进制数的形式自动装入定时器字中，这一点与“直接表示法”不一样，要注意。

在语句表中，还可以使用 IEC 格式的时间值，即在时间值的前面加 T#，例如 T#50S。

6. 2. 2　定时器指令分类

图 6-19 给出了 S7 -300 PLC 中五种类型定时器的工作状态时序图。

（1）脉冲定时器（SP，Pulse Timer）

如果在启动（S）输入端有一个上升沿，S_PULSE（脉冲 S5 定时器）将启动指定的定时器。信号变化始终是启用定时器的必要条件。定时器在输入端 S 的信号状态为“1”时运

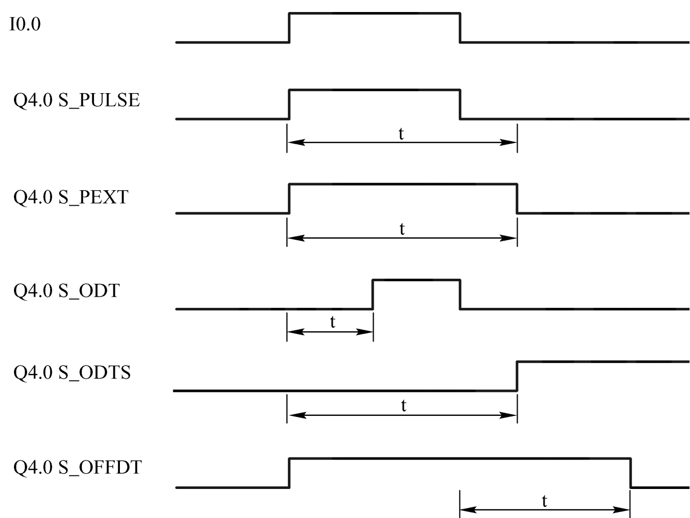


图 6-19　五种类型定时器的时序

行，但最长周期是由输入端 TV 指定的时间值。只要定时器运行，输出端 Q 的信号状态就为“1”。如果在时间间隔结束前，S 输入端从“1”变为“0”，则定时器将停止。在这种情况下，输出端 Q 的信号状态为“0”。

如果在定时器运行期间定时器复位（R）输入从“0”变为“1”时，则定时器将被复位。当前时间和时间基准也被设置为零。如果定时器不是正在运行，则定时器 R 输入端的逻辑“1”没有任何作用。

可在输出端 BI 和 BCD 扫描当前时间值。时间值在 BI 端是二进制编码，在 BCD 端是 BCD 编码。当前时间值为初始 TV 值减去定时器启动后经过的时间。

脉冲定时器时序波形图如图 6-20 所示。

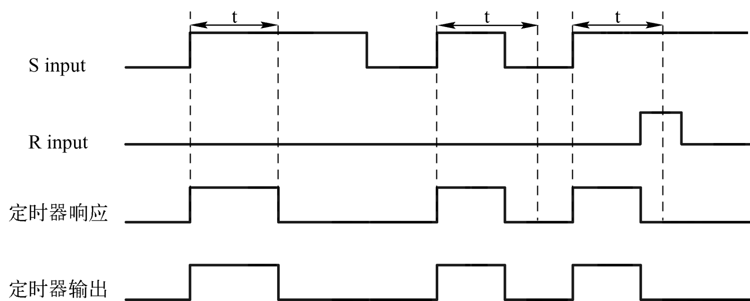


图 6-20　脉冲定时器时序波形图

在 STEP 7 中的使用如图 6-21 所示。

如果输入端 I1.0 的信号状态从“0”变为“1”（RLO 中的上升沿），则定时器 T10 将启动。只要 I1.0 为“1”，定时器就将继续运行指定的 10 s 时间。如果定时器达到预定时间前，I1.0 的信号状态从“1”变为“0”，则定时器将停止。如果输入端 I1.1 的信号状态从“0”变为“1”，而定时器仍在运行，则时间复位。只要定时器运行，输出端 Q5.0 就是逻辑“1”，如果定时器预设时间结束或复位，则输出端 Q5.0 变为“0”。MW10、MW20 为时间的当前值。

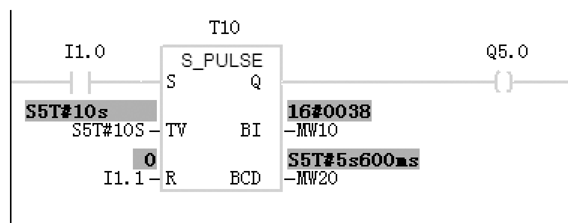


图 6-21　脉冲定时器（SP）

图 6-21 对应的 STL 语句表如下：

```
A     I1. 0
L     S5T#10s        \\装入定时时间到 ACCU1
SP    T10            \\启动脉冲定时器 T10
A     I1. 1
R     T10            \\定时器 T10 复位
L     T 10           \\装载当前时间值(整数格式)
T     MW 10
LC    T 10           \\装载当前时间值(BCD 格式)
T     MW 20
A     T10
=     Q5. 0
```

（2）扩展脉冲定时器（SE，Extended Pulse Timer）

如果在启动（S）输入端有一个上升沿，S_PEXT（扩展脉冲 S5 定时器）将启动指定的定时器。信号变化始终是启用定时器的必要条件。定时器以在输入端 TV 指定的预设时间间隔运行，即使在时间间隔结束前，S 输入端的信号状态变为“0”。只要定时器运行，输出端 Q 的信号状态就为“1”。如果在定时器运行期间输入端 S 的信号状态从“0”变为“1”，则将使用预设的时间值重新启动（“重新触发”）定时器。

如果在定时器运行期间复位（R）输入从“0”变为“1”，则定时器复位。当前时间和时间基准被设置为零。

可在输出端 BI 和 BCD 扫描当前时间值。时间值在 BI 处为二进制编码，在 BCD 处为 BCD 编码。当前时间值为初始 TV 值减去定时器启动后经过的时间。

扩展脉冲定时器时序波形图如图 6-22 所示。

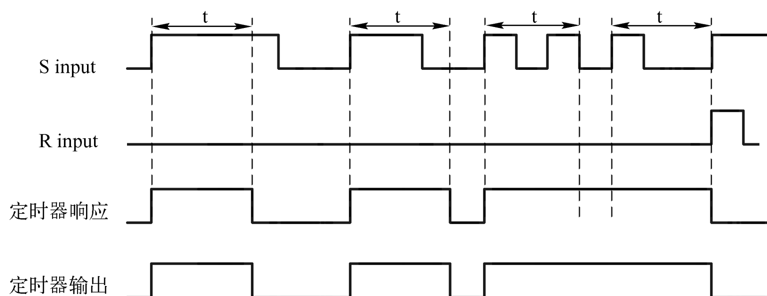


图 6-22　扩展脉冲定时器时序波形图

在 STEP 7 中使用如图 6-23 所示。

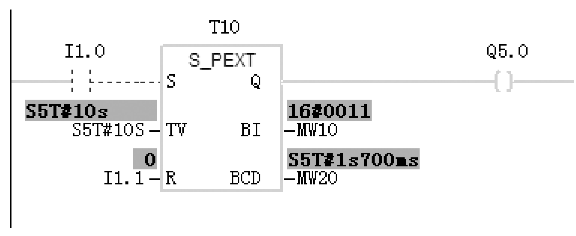


图 6-23　扩展脉冲定时器（SE）

如果输入端 I1.0 的信号状态从“0”变为“1”（RLO 中的上升沿），则定时器 T10 将启动。定时器将继续运行指定的 10 s 时间，而不会受到输入端 S 处下降沿的影响。如果在定时器达到预定时间前，I1.0 的信号状态从“0”变为“1”，则定时器将被重新触发。只要定时器运行，输出端 Q4.0 就为逻辑“1”。MW10、MW20 为时间的当前值。

图 6-23 对应的 STL 语句表如下：

```
A     I1.0
L     S5T#10s        \\装入定时时间到 ACCU1
SE    T10            \\启动扩展脉冲定时器 T10
A     I1.1
R     T 10           \\定时器 T10 复位
L     T 10           \\装载当前时间值(整数格式)
T     MW 10
LC    T 10           \\装载当前时间值(BCD 格式)
T     MW 20
A     T10
=     Q5.0
```

（3）接通延时定时器（SD，ON－Delay Timer）

接通延时定时器是使用最多的定时器。如果在启动（S）输入端有一个上升沿，S_ODT（接通延时 S5 定时器）将启动指定的定时器。信号变化始终是启用定时器的必要条件。只要输入端 S 的信号状态为“1”，定时器就以在输入端 TV 指定的时间间隔运行。定时器达到指定时间而没有出错，并且 S 输入端的信号状态仍为“1”时，输出端 Q 的信号状态为“1”。如果定时器运行期间输入端 S 的信号状态从“1”变为“0”，定时器将停止，在这种情况下，输出端 Q 的信号状态为“0”。

如果在定时器运行期间复位（R）输入从“0”变为“1”，则定时器复位。当前时间和时间基准被设置为零。然后，输出端 Q 的信号状态变为“0”。如果在定时器没有运行时 R 输入端有一个逻辑“1”，并且输入端 S 的 RLO 为“1”，则定时器也复位。

可在输出端 BI 和 BCD 扫描当前时间值。时间值在 BI 处为二进制编码，在 BCD 处为 BCD 编码。当前时间值为初始 TV 值减去定时器启动后经过的时间。

接通延时定时器时序波形图如图 6-24 所示。

在 STEP 7 中的使用如图 6-25 所示。

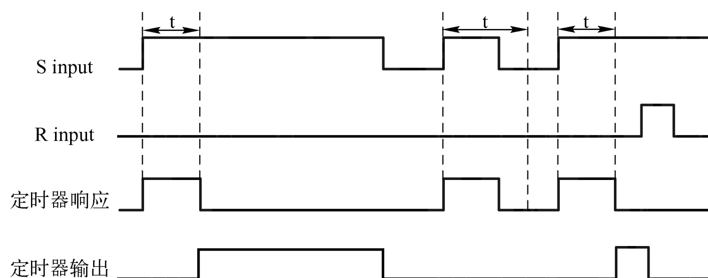


图 6-24　接通延时定时器时序波形图

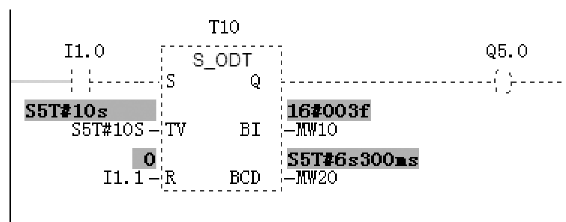


图 6-25　接通延时定时器（SD）

如果 I1. 0 的信号状态从“0”变为“1”（RLO 中的上升沿），则定时器 T10 将启动。如果指定的 10 s 时间结束并且输入端 I1. 0 的信号状态仍为“1”，则输出端 Q5. 0 将为“1”。如果 I1. 0 的信号状态从“1”变为“0”，则定时器停止，并且 Q5. 0 将为“0”（如果 I1. 1 的信号状态从“0”变为“1”，则无论定时器是否运行，时间都复位）。MW10、MW20 为时间的当前值。

图 6-25 对应的 STL 语句表如下：

```
A     I 1.0
L     S5T#10s      \\装入定时时间到 ACCU1
SD    T10          \\启动接通延时定时器 T10
A     I1. 1
R     T 10         \\定时器 T10 复位
L     T 10         \\装载当前时间值(整数格式)
T     MW 10
LC    T 10         \\装载当前时间值(BCD 格式)
T     MW 20
A     T10
=     Q5. 0
```

（4）带保持的接通延时定时器（SS，Retentive ON－Delay Timer）

如果在启动（S）输入端有一个上升沿，S_ODTS（保持接通延时 S5 定时器）将启动指定的定时器。信号变化始终是启用定时器的必要条件。定时器以在输入端 TV 指定的时间间隔运行，即使在时间间隔结束前，输入端 S 的信号状态变为“0”。定时器预定时间结束时，输出端 Q 的信号状态为“1”，而无论输入端 S 的信号状态如何。如果在定时器运行时输入端 S 的信号状态从“0”变为“1”，则定时器将以指定的时间重新启动（重新触发）。

如果复位（R）输入从“0”变为“1”，则无论S输入端的RLO如何变化，定时器都将复位。然后，输出端Q的信号状态变为“0”。

可在输出端BI和BCD扫描当前时间值。时间值在BI端是二进制编码，在BCD端是BCD编码。当前时间值为初始TV值减去定时器启动后经过的时间。

带保持的接通延时定时器时序波形图如图6-26所示。

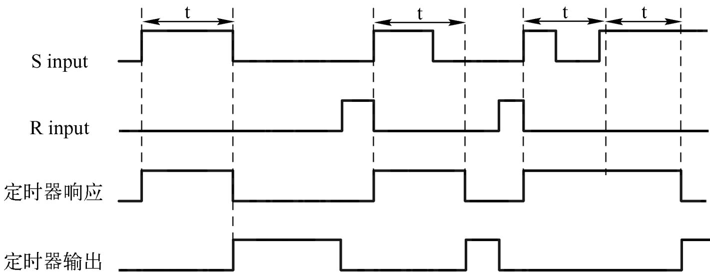


图6-26　带保持的接通延时定时器时序波形图

其在STEP 7中的使用如图6-27所示。

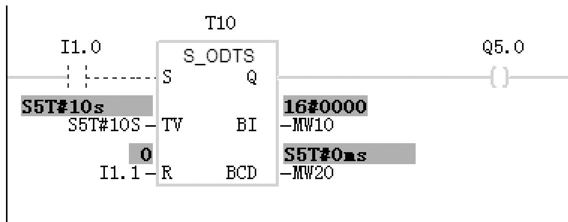


图6-27　带保持的接通延时定时器（SS）

如果I1.0的信号状态从“0”变为“1”（RLO中的上升沿），则定时器T10将启动。无论I1.0的信号是否从“1”变为“0”，定时器都将运行。如果在定时器达到指定时间前，I1.0的信号状态从“0”变为“1”，则定时器将重新触发。如果定时器达到指定时间，则输出端Q5.0将变为“1”。如果输入端I1.1的信号状态从“0”变为“1”，则无论S处的RLO如何，时间都将复位。MW10、MW20为时间的当前值。

图6-27对应的STL语句表如下：

```
A     I 1.0
L     S5T#10s          \\装入定时时间到ACCU1
SS    T10              \\启动保持型接通延时定时器器T10
A     I1.1
R     T 10             \\定时器T10复位
L     T 10             \\装载当前时间值(整数格式)
T     MW 10
LC    T 10             \\装载当前时间值(BCD格式)
T     MW 20
A     T10
=     Q5.0
```

（5）断电延时定时器（SF，Off - Delay Timer）

如果在启动（S）输入端有一个下降沿，S_OFFDT（断开延时 S5 定时器）将启动指定的定时器。信号变化始终是启用定时器的必要条件。如果 S 输入端的信号状态为“1”，或定时器正在运行，则输出端 Q 的信号状态为“1”。如果在定时器运行期间输入端 S 的信号状态从“0”变为“1”时，定时器将复位。输入端 S 的信号状态再次从“1”变为“0”后，定时器才能重新启动。

如果在定时器运行期间复位（R）输入从“0”变为“1”时，定时器将复位。

可在输出端 BI 和 BCD 扫描当前时间值。时间值在 BI 端是二进制编码，在 BCD 端是 BCD 编码。当前时间值为初始 TV 值减去定时器启动后经过的时间。

断电延时定时器时序波形图如图 6-28 所示。

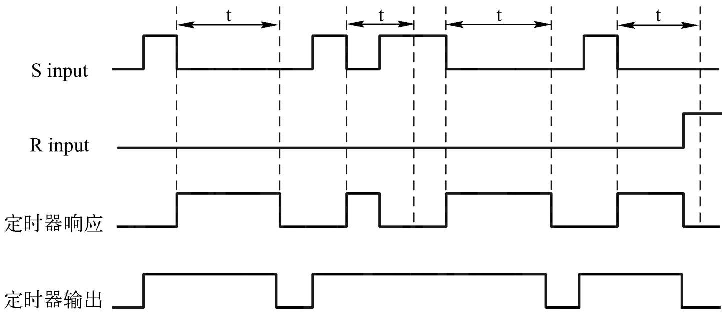


图 6-28　接通延时定时器时序波形图

在 STEP 7 中的使用如图 6-29 所示。

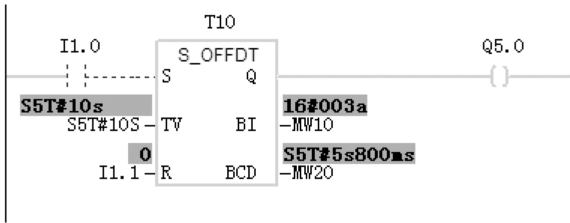


图 6-29　断电延时定时器（SF）

如果 I1.0 的信号状态从“1”变为“0”，则定时器启动。I1.0 为“1”或定时器运行时，Q5.0 为“1”。如果在定时器运行期间 I1.1 的信号状态从“0”变为“1”，则定时器复位。MW10、MW20 为时间的当前值。

图 6-29 对应的 STL 语句表如下：

```
A     I 1.0
L     S5T#10s       \\装入定时时间到 ACCU1
SF    T10           \\启动断电延时定时器器 T10
A     I1.1
R     T 10          \\定时器 T10 复位
L     T 10          \\装载当前时间值(整数格式)
T     MW 10
```

```
LC    T 10        \\装载当前时间值(BCD 格式)
T     MW 20
A     T10
=     Q5.0
```

（6）定时器线圈指令

S7-300 PLC 中有五种类型的定时器还有其线圈指令形式可以使用，见表 6-9。线圈指令的使用可以简化梯形图结构，很方便地来组织程序。定时器线圈指令与定时器功能指令的区别是定时器线圈指令不能检测瞬时定时值。

表 6-9　五种类型定时器线圈指令

LAD 指令	STL 指令	功　能
T no. ——(SP) 时间值	SP　T no.	启动脉冲定时器 时间值的数据类型为： S5TIME
T no. ——(SE) 时间值	SE　T no.	启动扩展脉冲定时器
T no. ——(SD) 时间值	SD　T no.	启动接通延时定时器
T no. ——(SS) 时间值	SS　T no.	启动保持型接通延时定时器
T no. ——(SF) 时间值	SF　T no.	启动关断延时定时器
	FR T no.	允许再启动定时器

线圈指令的使用如图 6-30 所示。

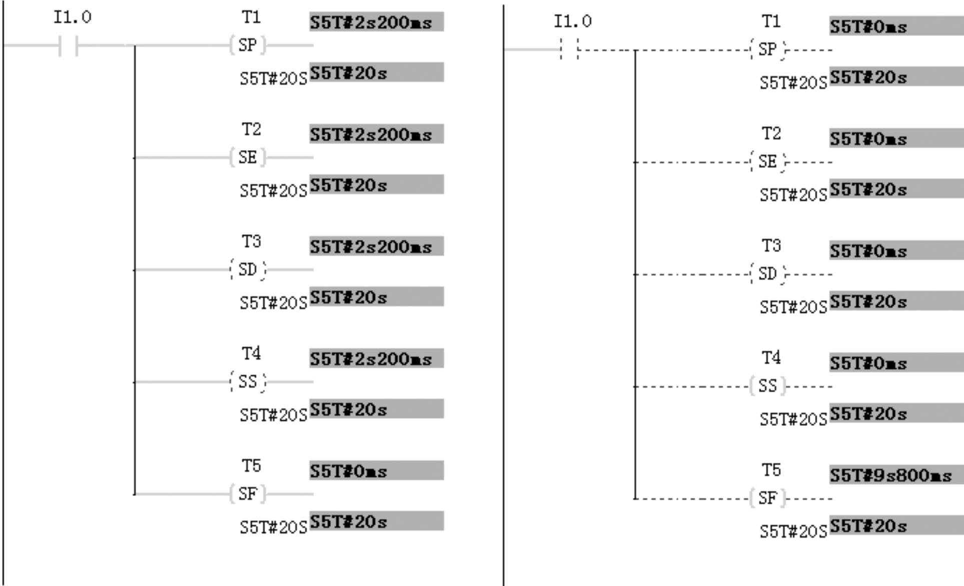


图 6-30　定时器线圈指令

比较定时器线圈和定时器方块指令不难看出：方块指令中用 TV 端可直接进行定时时间设定（只能用 S5TIME 格式）；用 Q 端可直接进行定时器对外输出；定时器的剩余定时时间可分别用二进制数和 BCD 数从 BI 端和 BCD 端输出，方便用户使用及查看。

对于以上 5 种不同形式的定时器指令，一般的选择原则如下。

如果要求输出信号为 1 的时间等于定时器的设定时间，且要求输入与输出信号状态一致时，可选择脉冲定时器 SP。

如果要求输出信号为 1 的时间等于定时器的设定时间，但不要求输入与输出信号状态一致，不考虑输入信号为 1 的时间长短，可选择扩展脉冲定时器 SE。

如果要求设定时间到且输入信号仍为 1 时，输出信号才从 0 变到 1，可选择接通延时定时器 SD。

如果要求设定时间到时，输出信号才从 0 变到 1，而不考虑输入信号此时的状态即为 1 的时间长短，可选择保持型接通延时定时器 SS。

如果要求输入信号从 0 变化到 1 时，输出信号也从 0 变化到 1，当输入信号从 1 变化到 0 时才开始延时，延时时间到时输出信号才从 1 变化到 0，可选择关断延时定时器 SF。

同时 S7－300 PLC 的定时器种类比较多，在编程时恰当使用可以简化程序，并且对于定时时间很短或者精度要求比较高时，需要采用外部时钟来实现。

6.2.3　定时器举例

【例 6-5】用接通延时定时器设计周期与占空比可调的振荡电路。

分析：一个周期与占空比可调振荡电路在 PLC 中实现，要用脉冲，而脉冲的占空比用计时器来实现。考虑到脉冲的一个周期有“0”和“1”两个状态，要用到两个计时器来完成计周期与占空比可调的脉冲。具体程序实现如图 6-31 所示。

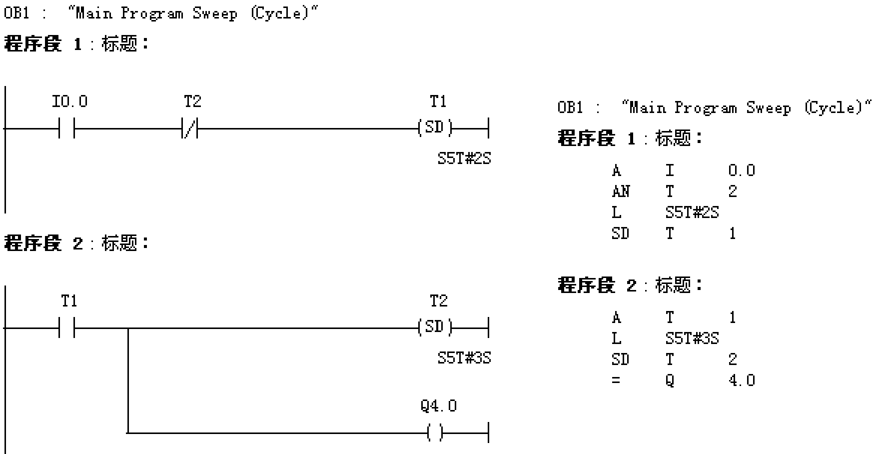


图 6-31　振荡电路梯形图及语句表

程序功能分析：当 I0.0 被按下，计时器 T1 开始工作，2 s 后，T1 接通，T2 开始工作，再过 3 s，T2 接通；下一个扫描周期，T1 停止工作，接下来 T2 停止工作；再下一个扫描周期，T1 又开始工作，如此往复。Q5.0 将以占空比 3∶5 的比例输出脉冲。

【例 6-6】用脉冲定时器设计一个周期振荡电路，振荡周期为 15 s，占空比为 1∶3。

说明：在设计中，用 T1 和 T2 分别定时 10 s 和 5 s，用 I0.0 启动振荡电路。由于是周期振荡电路，所以 T1 和 T2 必须互相启动。具体程序实现如图 6-32 所示。在 Network1 中，T2 需用常闭触点，否则，T1 无法启动。在 Network2 中，T1 工作期间，T2 不能启动工作。所以 T1 需用常闭触点来启动 T2。即当 T1 定时时间到时，T1 的常闭触点断开，从而产生 RLO 上跳沿，启动 T2 定时器。如此循环，在 Q4.0 端形成振荡电路。

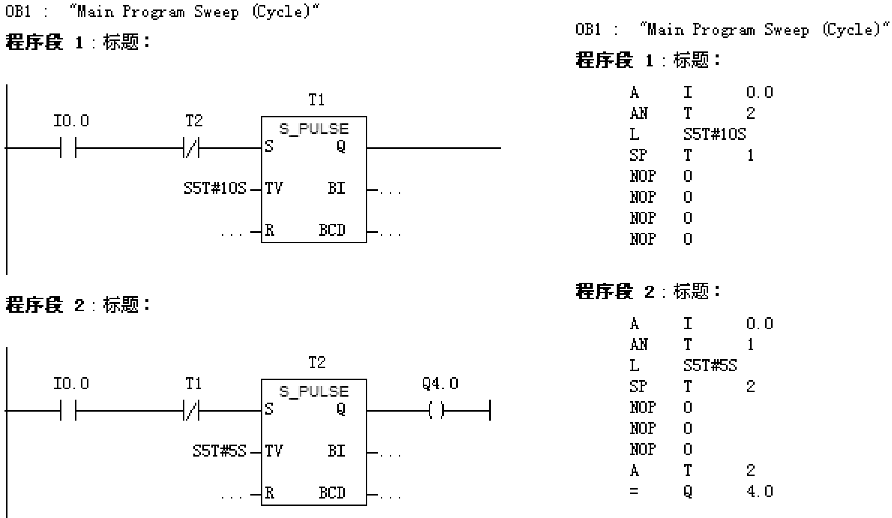


图 6-32　振荡电路梯形图及语句表

【例 6-7】周期性的脉冲也可以通过 CPU 的硬件来产生。在 S7 系列 PLC 的 CPU 的位存储器 M 中，可以任意指定一个字节，如 MB100，作为时钟脉冲存储器，当 PLC 运行时，MB100 的各个位，能周期性地改变二进制值，即产生不同频率（或周期）的时钟脉冲。要使用该功能，在硬件配置时需要设置 CPU 的属性，在硬件组态中双击 CPU 所在的槽，在其弹出的属性对话框中，选中"周期/时钟存储器"，其中有一个选项为时钟存储器，选中选择框就可激活该功能。如图 6-33 所示。

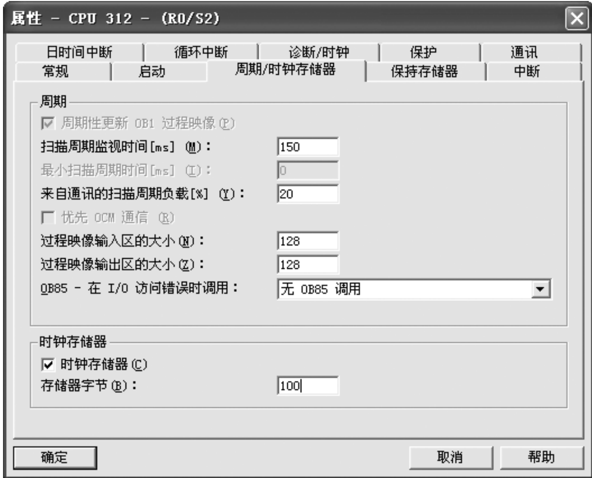


图 6-33　周期性脉冲设置

Clock Memory 各位的周期及频率见表 6-10。

表 6-10　Clock Memory 各位的周期及频率表

位	7	6	5	4	3	2	1	0
周期/s	2	1.6	1	0.8	0.5	0.4	0.2	0.1
频率/Hz	0.5	0.625	1	1.25	2	2.5	5	10

【例 6-8】 用计数器扩展定时器的定时范围。要求：I0.0 为复位按钮兼启动按钮，定时范围为 12 h。12 h 之后，将电磁阀 Q4.0 打开。

分析：单个定时器最长的时间是 9990 s，约 2 个多小时。为了实现 12 h 的定时功能，可以先设计一周期振荡电路，其中接通延时定时器 T1 和 T2 的定时时间均为 7200 s，这样振荡周期为 4 h，如果结合一个初始值为 3 的减法计数器，每隔 4 h 触发，则在减计数器计数值减至零时，相当于经过了 12 h。其梯形图如图 6-34 所示。

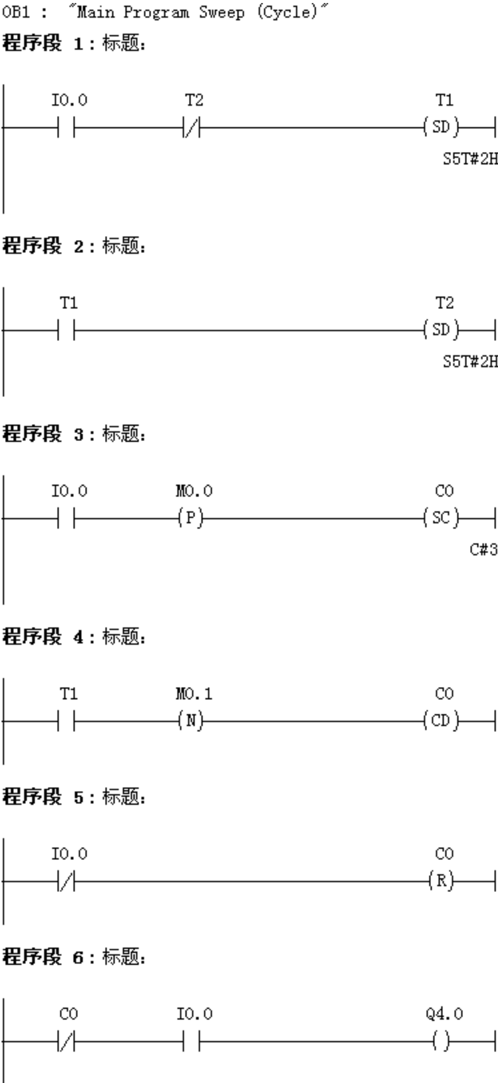


图 6-34　定时器扩展程序图

6.3 计数器指令

6.3.1 计数器简介

（1）计数器基本知识

计数器用于对计数器指令前面程序的逻辑操作结果 RLO 的正跳沿（即正脉冲）计数。计数器又分普通计数器和高速计数器，本节将对普通计数器进行讲解。计数器是一种由位和字组成的复合单元，计数器的输出由位表示，其计数值存储在字存储器中。在 CPU 的存储器中留出了计数器区域，该区域用于存储计数器的计数值。每个计数器为两个字节（Byte），称为计数字。计数器字中的第 0～11 位表示计数值（BCD 码格式），三位 BCD 码表示的计数范围是 0～999，第 12～15 位没有用途。当计数器“加计数”达到上限 999 时，累加停止（即 999+1=999）；“减计数”达到 0 时，将不再减少（即 0－1=0）。计数器地址就是“C〈元件号〉”，如 C1、C20 等。

不同的 CPU 模块，用于计数器的存储区域也不同，最多允许使用 64～512 个计数器。计数器字格式如图 6-35 所示。

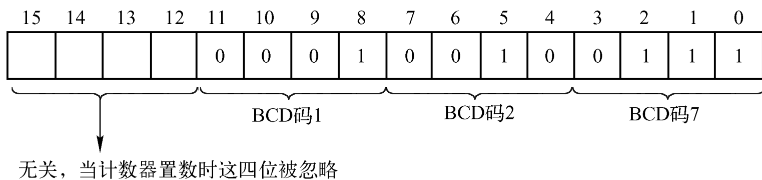


图 6-35　计数器字格式

当计数器启动时，累加器 1 低字的内容被当作计数初值装入计数字中。这一过程是由操作系统控制自动完成的，用户只需给累加器 1 装入不同的数值，即可设置需要的计数初值。只要计数器的计数值不是“0”，计数器的输出就为“1”。

```
L      C# xyz
```

其中：xyz = 记数初值，取值范围为 1～999。

（2）计数器动作过程

在其他型号的 PLC 中，甚至是德国西门子的 S7－200PLC，计数器的设定值是与“计数到”的概念相关联的。也就是说，在常规中，当计数达到设定值时，计数器输出触点（即计数器的位）有动作。但 S7－300PLC 的计数器与此不同，只要“当前计数值”不为 0，计数器的输出为 1，即其常开触点闭合，常闭触点打开。

然而，“计数到，计数器输出有动作”的概念在生产过程控制中是经常用到的，可 S7－300PLC 的计数器却不符合这一概念，即不符合常规。它常用以下两种方法来实现“计数到”。

① 减法计数器

先把设定的计数初值送入计数器字中，计数器输出便立刻从 0 到 1，产生一个正跳变沿。在“当前计数值”大于 0 的时候，计数器输出为 1；当减计数减到 0，即“当前计数

值”等于0时，计数器输出从1到0，产生一个负跳变沿，再用负跳变沿检测指令，测出计数器“计数到”，也可以用其他方法检测“计数到”，例如，用计数器的常闭触点与装计数值指令的允许信号的常开触点串联也可测出计数器“计数到”。

② 加法计数器

置计数初值时，计数器输出不动作，输出为0。在“当前计数值”大于0的时候，其输出为1（实际上，加法计数器工作时，计数值总是大于0，输出总为1，只有当复位时，输出才为0）。若加计数加到大于或等于计数初值时，其输出仍为1，不变化，此时可用查看“当前剩余计数值（BCD数）”指令，即“LC C〈元件号〉”查出计数器的“当前计数值”，再用装入指令“T〈指定字地址〉”把当前计数值转移到“该指定的字地址”上去，最后用“比较指令”把当前计数值与设定的计数初置（常数）进行比较，若相等，则说明“计数到”，比较指令的结果（相当于一个特殊触点）输出为1，相当于“计数到”时计数器输出从0到1，满足了常规的情况。

综上所述，无论是加法计数器还是减法计数器，只要当前计数值等于0，计数器输出为0；若当前计数值大于0，其输出为1，复位时，计数值清零，其输出为0。

总之，加法计数器使用起来比较麻烦，而减法计数器则相对简便，故在S7－300PLC中常用减法计数器。

6.3.2 计数器指令分类

在STEP 7中计数器有图6-36所示的三种指令。

在STEP 7中除了块图形式的计数器指令以外，S7－300 PLC系统还为用户准备了LAD环境下的线圈形式的计数器。这些指令有计数器初值预置指令SC、加计数器指令CU和减计数器指令CD。见表6-11。

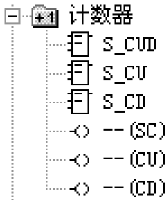


图6-36 STEP 7中计数器指令

表6-11 计数器线圈指令

LAD	参数	数据类型	存储区	描述
<C编号> －－－(SC)－－－ <预设值>	C编号	COUNTER	C	计数器标识号，其范围依赖于CPU，地址表示预置初值的计数器号
	预置值	WORD	I、Q、M、L、D	预置值（必须为BCD码格式，即为C#，例如C#30）
<C编号> －－－(CU)	C编号	COUNTER	C	地址表示要执行加法计数的计数器号
<C编号> －－－(CD)	C编号	COUNTER	C	地址表示要执行减法计数的计数器号

当逻辑位RLO有正跳沿时，计数器置初值线圈将预置值装入指定计数器中。若RLO位的状态没有正跳沿发生，则计数器的值保持不变。

初值预置SC指令若与CU指令配合可实现S_CU指令的功能，当逻辑位RLO有正跳沿时，加法计数器线圈使指定计数器的值加1，如果RLO位的状态没有正跳沿发生，或者计数器数值已经达到最大值999，则计数器的值保持不变。

SC 指令若与 CD 指令配合可实现 S_CD 指令的功能，当逻辑位 RLO 有正跳沿时，减法计数器线圈使指定计数器的值减 1，如果 RLO 位的状态没有正跳沿发生，或者计数器数值已经达到最小值 0，则计数器的值保持不变。SC 指令若与 CU 和 CD 配合可实现 S_CUD 的功能。

（1）加计数器指令（S_CU，Up Counter）

加计数器梯形图指令的端子说明见表 6-12。

表 6-12　加计数器

S_CU 加计数器	参　　数	数据类型	存　储　区	描　　述
C no. S_CU CU　Q … S　CV … … PV　CV_BCD … … R	C 编号	COUNTER	C	计数器标识号，其范围依赖于 CPU
	CU	BOOL	I、Q、M、L、D	加计数输入
	S	BOOL	I、Q、M、L、D	为预设计数器设置输入
	PV	WORD	I、Q、M、L、D 或常数	将计数器值以“C# < 值 >”的格式输入（范围 0～999）
	R	BOOL	I、Q、M、L、D	复位输入
	CV	WORD	I、Q、M、L、D	当前计数器值，十六进制
	CV_ BCD	WORD	I、Q、M、L、	当前计数器值，BCD 码
	Q	BOOL	I、Q、M、L、D	计数器状态输出

如果输入 S 有上升沿，则 S_CU（加计数器）预置为输入 PV 的值。

如果输入 R 为“1”，则计数器复位，并将计数值设置为零。

如果输入 CU 的信号状态从“0”切换为“1”，并且计数器的值小于“999”，则计数器的值增 1。

如果已设置计数器并且输入 CU 为 RLO = 1，则即使没有从上升沿到下降沿或下降沿到上升沿的变化，计数器也会在下一个扫描周期进行相应的计数。

如果计数值大于等于零（“0”），则输出 Q 的信号状态为“1”。

（2）减计数器指令（S_CD，Down Counter）

减计数器梯形图指令的端子说明见表 6-13。

表 6-13　减计数器

S_CD 加计数器	参　　数	数据类型	存　储　区	描　　述
C no. S_CD CD　Q … S　CV … … PV　CV_BCD … … R	C 编号	COUNTER	C	计数器标识号，其范围依赖于 CPU
	CD	BOOL	I、Q、M、L、D	降值计数输入
	S	BOOL	I、Q、M、L、D	为预设计数器设置输入
	PV	WORD	I、Q、M、L、D 或常数	将计数器值以“C# < 值 >”的格式输入（范围 0～999）
	R	BOOL	I、Q、M、L、D	复位输入
	CV	WORD	I、Q、M、L、D	当前计数器值，十六进制
	CV_BCD	WORD	I、Q、M、L、D	当前计数器值，BCD 码
	Q	BOOL	I、Q、M、L、D	计数器状态

如果输入 S 有上升沿，则 S_CD（减计数器）设置为输入 PV 的值。

如果输入 R 为 1，则计数器复位，并将计数值设置为零。

如果输入 CD 的信号状态从“0”切换为“1”，并且计数器的值大于零，则计数器的值减 1。

如果已设置计数器并且输入 CD 为 RLO = 1，则即使没有从上升沿到下降沿或下降沿到上升沿的变化，计数器也会在下一个扫描周期进行相应的计数。

如果计数值大于等于零（“0”），则输出 Q 的信号状态为“1”。

减计数器的使用如图 6-37 所示。

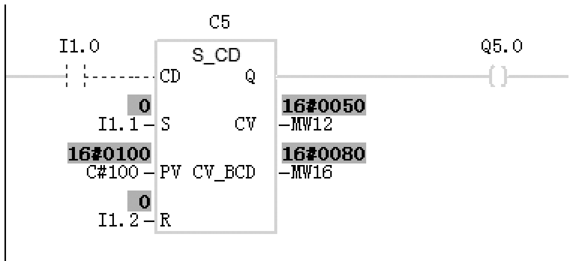


图 6-37　减计数器

如果 I1.1 从“0”改变为“1”，则计数器预置为 MW12、MW16 的值。如果 I1.0 的信号状态从“0”改变为“1”，则计数器 C5 的值将减 1，当 C5 的值等于“0”时除外。如果 C10 不等于零，则 Q5.0 为“1”。

（3）加减计数器（S_CUD）

加减计数器梯形图指令的端子说明见表 6-14。

表 6-14　加减计数器

S_CUD 加计数器	参　　数	数 据 类 型	存　储　区	描　　述
C no. S_CUD CU　Q … CD　CV … … S　CV_BCD … PV … R	C 编号	COUNTER	C	计数器标识号，其范围依赖于 CPU
	CU	BOOL	I、Q、M、L、D	加计数输入
	CD	BOOL	I、Q、M、L、D	减计数输入
	S	BOOL	I、Q、M、L、D	为预设计数器设置输入
	PV	WORD	I、Q、M、L、D 或常数	将计数器值以“C# < 值 >”的格式输入（范围 0 至 999）
	R	BOOL	I、Q、M、L、D	复位输入
	CV	WORD	I、Q、M、L、D	当前计数器值，十六进制
	CV_BCD	WORD	I、Q、M、L、D	当前计数器值，BCD 码
	Q	BOOL	I、Q、M、L、D	计数器状态

如果输入 S 有上升沿，S_CUD（双向计数器）预置为输入 PV 的值。如果输入 R 为 1，则计数器复位，并将计数值设置为零。如果输入 CU 的信号状态从“0”改变为“1”，并且计数器的值小于“999”，则计数器的值增 1。如果输入 CD 有上升沿，并且计数器的值大于“0”，则计数器的值减 1。

如果两个计数输入都有上升沿，则执行两个指令，并且计数值保持不变。

如果已设置计数器并且输入 CU/CD 为 RLO = 1，即使没有从上升沿到下降沿或下降沿到上升沿的变化，计数器也会在下一个扫描周期进行相应的计数。

加减计数器的使用如图 6-38 所示。

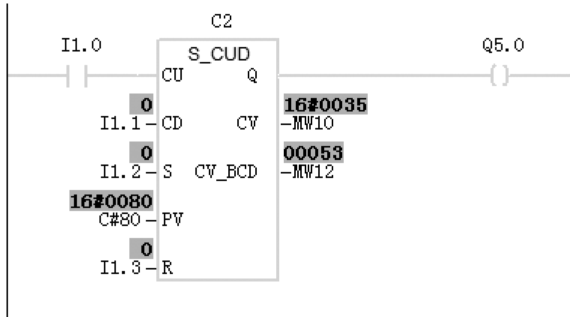


图 6-38　加减计数器

如果 I1.2 从“0”改变为“1”，则计数器预置为 C#80，十六进制为 16#0050。如果 I1.0 的信号状态从“0”改变为“1”，则计数器 C2 的值将增加 1 ，当 C2 的值等于“999”时除外。如果 I1.1 从“0”改变为“1”，则 C2 减少 1 ，但当 C10 的值为“0”时除外。如果 C2 不等于零，则 Q5.0 为“1”。

（4）计数器使用注意事项

① 计数脉冲从何而来，即计数器的启动问题；同时注意：计数器指令的加、减计数输入端以及预置值输入端均为上升沿执行，即逻辑位必须有从“0”到“1”的变化时，指令才会执行。

② 在开始动作之前，需要计多少个数，即赋值问题。比如要将 10 个货物装入一个箱子中，那就要赋值为 10，并使用减计数器。

③ 如何复位计数器，让它重新开始计数。比如一个箱子装满 10 个货物后，需要再装另外一个箱子，此时必须重新启动计数器。

④ 计数器触点的状态由计数器的值决定，如果计数值等于零，则计数器触点的状态为低电平“0”，如果计数值不等于零（无论等于几），则计数器触点的状态为高电平“1”。

⑤ 注意现场监控当前计数值。

6.3.3　计数器举例

计数器编程顺序是：启动加计数或启动减计数→计数器置数→计数器复位→检测计数器输出状态。

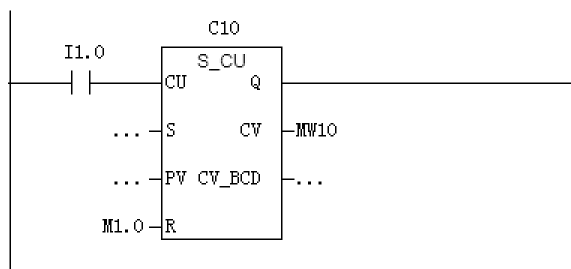
【例 6-9】用比较和计数指令编写开关灯程序，要求灯控按钮 I1.0 按下一次，灯 Q5.0 亮，按下两次，灯 Q5.0、Q5.1 全亮，按下三次灯全灭，如此循环。

分析：在程序中所用计数器为加法计数器，当加到 3 时，必须复位计数器，这是关键。灯控制程序如图 6-39 所示。

【例 6-10】在 S7-300 中，单个计数器的最大计数值是 999，如果要求大于 999 的计数，就要进行扩展。结合应用传送指令和比较指令，将两个计数器级连，最大计数值可达 999^2，n 个计数器级连，最大计数值可达 999^n。程序如图 6-40 所示。

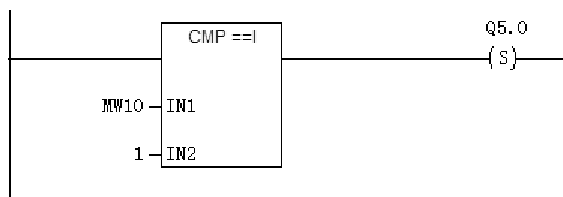
程序段 1：标题：

注释：



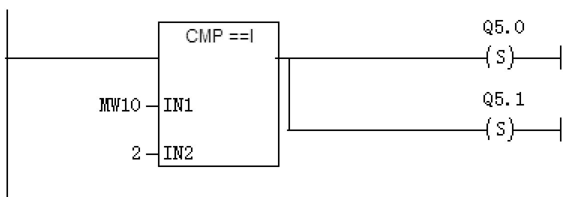
程序段 2：标题：

注释：



程序段 3：标题：

注释：



程序段 4：标题：

注释：

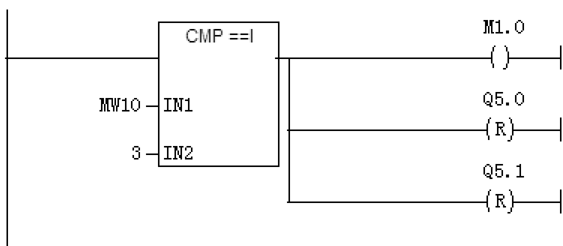


图 6–39　开关灯程序梯形图

程序段 1：标题：

注释：



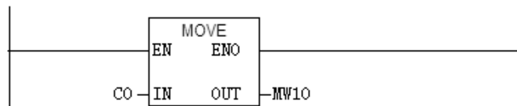
程序段 2：标题：

注释：



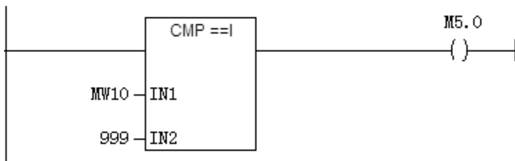
程序段 3：标题：

注释：



程序段 4：标题：

注释：



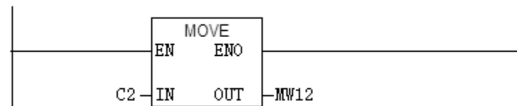
程序段 5：标题：

注释：



程序段 6：标题：

注释：



程序段 7：标题：

注释：

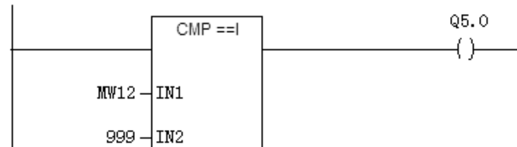


图 6–40　计数器扩展梯形图

6.4　数据处理指令

6.4.1　装入指令与传送指令

（1）装入指令

装入 L（Load）和传送 T（Transfer）指令可以在存储区之间或存储区与过程输入、输出之间交换数据。CPU 在每次扫描中无条件地执行这些指令，这些指令的执行不受逻辑操作结果 RLO 状态的影响。L 指令将源操作数装入累加器 1 中，而累加器原有的数据移入累加器 2 中，累加器 2 中原有的内容被覆盖。T 指令将累加器 1 中的内容写入目的存储区中，累加器的内容保持不变。L 和 T 指令可对字节（8 位）、字（16 位）、双字（32 位）数据进行操作，当数据长度小于 32 位时，数据在累加器右对齐（低位对齐），其余各位填 0。基本的装入和传送指令如下：

```
L    10               //将立即数 10 装入累加器 1 中
L    MW 20            //将 MW20 中的值装入累加器 1 中
L    IB[DID 10]       //将由数据双字 DID10 指出的输入字节装入累加器 1 中
T    MW 20            //将累加器 1 中的内容传送给存储字 MW20
T    MW[AR1,P#20.0]   //将累加器 1 中的内容传送给由地址寄存器 1 加偏移量
                      //确定的存储字中
```

（2）读取或传送状态字

L STW（将状态字中 0 ~ 8 位装入累加器 1 中，累加器 9 ~ 31 位被清 0），将状态字装入累加器 1 中，指令的执行与状态位无关，而且对状态字没有任何影响。

T STW（将累加器 1 的内容传送到状态字），使用 T STW 指令可以将累加器 1 的位 0 ~ 8 传送到状态字的相应位，指令的执行与状态位无关。

注意：对 S7 - 300 系列 PLC 的 CPU，L　STW 不对状态字中的 SC、STA、OR 位进行操作，仅将状态字中的 1、4、5、6、7 位装入累加器 1 的对应位。

（3）装入时间值或计数值

定时器字中的剩余时间值以二进制格式保存，用 L 指令从定时器字中读出二进制时间值装入累加器 1 中，称为直接装载。也可用 LC 指令以 BCD 码格式读出时间值，装入累加器 1 低字中，称为 BCD 码格式读出时间值。以 BCD 码格式装入时间值可以同时获得时间值和时基，时基与时间值相乘就得到定时剩余时间。

```
L    T2    //将定时器 T2 中二进制格式的时间值直接装入累加器 1 的低字中
LC   T2    //将定时器 T2 的时间值和时基以 BCD 码装入累加器 1 的低字中
```

对当前计数值也可以直接装载和以 BCD 码格式读出当前计数值。

```
L    C5    //将计数器 C5 中二进制格式的计数值直接装入累加器 1 的低字中
LC   C5    //将计数器 C5 中的计数值以 BCD 码格式装入累加器 1 的低字中
```

（4）地址寄存器装入和传送

对于地址寄存器，可以不经过累加器 1 而直接将操作数装入或传送，或将两个地址寄存器的内容直接交换。

指令格式：LAR1，说明：将操作数的内容装入地址寄存器 1（AR1）。

指令格式：LAR2，说明：将操作数的内容装入地址寄存器 2（AR2）。

指令格式：TAR1，说明：将 AR1 的内容传送给存储区或 AR2。

指令格式：TAR2，说明：将 AR1 的内容传送给存储区或 AR2。

指令格式：CAR，说明：交换 AR1 和 AR2 的内容。

下面的例子说明了指令的用法：

```
LAR1   P# I1.0     //将输入位 I1.0 的地址指针装入 AR1
LAR2   P# 0.0      //将二进制数 2#0000 0000 0000 0000 0000 0000 0000 0000 装入 AR2
LAR1   P#Stop      //将符号名为 Stop 的存储器的地址指针装入 AR1
LAR1   AR2         //将 AR2 的内容装入 AR1
LAR1   DBD 10      //将数据双字 DBD 10 的内容装入 AR1
TAR1   AR2         //将 AR1 的内容传送至 AR2
TAR2               //将 AR2 的内容传送至累加器 1
TAR1   MD 30       //将 AR1 的内容传送至存储器双字 MD 30
CAR                //交换 AR1 和 AR2 的内容
```

（5）传送指令

MOVE 指令为功能框形式的传送指令，能够复制字节、字或双字数据对象。应用中 IN 和 OUT 端操作数可以是常数、I、Q、M、D、L 等类型，数据类型必须匹配，其梯形图端子见表 6-15。

表 6-15　传送指令

MOVE 传送指令	参　数	数据类型	存储区	说　明
MOVE EN OUT IN ENO	EN	BOOL	I、Q、M、D、L	允许输入
	ENO	BOOL	Q、M、D、L	允许输出
	IN	8、16、32 位长的所有数据类型	I、Q、M、D、L	源数值（可为常数）
	OUT	8、16、32 位长的所有数据类型	Q、M、D、L	目的操作数

MOVE 指令通过启用 EN 输入来激活。在 IN 输入端指定的值将复制到在 OUT 输出端指定的地址。ENO 与 EN 的逻辑状态相同。MOVE 只能复制 BYTE、WORD 或 DWORD 数据对象。用户自定义数据类型（如数组或结构）必须使用系统功能“BLKMOVE”（SFC 20）来实现传输。其使用如图 6-41 所示。

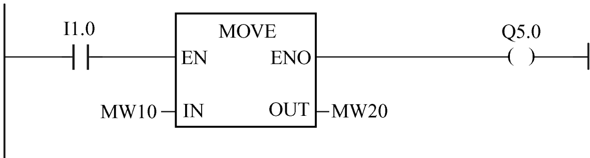


图 6-41　传送指令

当I1.0为“1”，则执行指令。把MW10的内容复制MW20中，MW10的数保持不变，Q5.0为“1”。实现上述相同功能的语句表指令为：

```
        A     I1.0
        JNB   _001        // 若RLO为0,则跳转到_001处
        L     MW 10       // 若RLO为1,则执行装载指令
        T     MW 20       // 将MW 10的内容送到MW 20中
        SET               // 使RLO为1
        SAVE              // 保存RLO,使BR为1
        CLR               // 清除RLO
  _001: A     BR
        =     Q 5. 0
```

注意：在为变量赋初始值时，为了保证传输只执行一次，一般MOVE方块指令和边缘触发指令联合使用。

6.4.2 比较指令

比较指令用于比较累加器2与累加器1中的数据大小。比较时应确保两个数的数据类型相同，数据类型可以是整数、长整数或实数．若比较的结果为真，则RLO为1，否则为0。比较指令影响状态字，用指令测试状态字有关位，可得到两个数更详细的情况。

在STEP 7中有3种类型、6种关系的比较指令，即大于、小于、等于、不等于、大于等于和小于等于。这6种关系的比较数有3种，即整数、双整数和实数。比较指令只能是相同的数据类型进行数据比较。

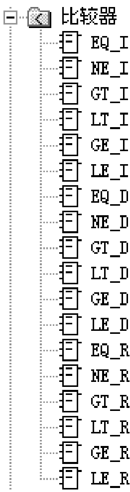


图6-42　STEP 7中比较指指令

在STEP 7中的比较指令如图6-42所示。

（1）整数比较指令

说明见表6-16。

表6-16　整数比较指令

LAD方块	STL指令	方块上部的符号	比 较 类 型
CMP==I IN1 IN2	==I	==	IN1等于IN2
	<>I	<>	IN1不等于IN2
	>I	>	IN1大于IN2
	<I	<	IN1小于IN2
	> =I	> =	IN1大于等于IN2
	< =I	< =	IN1小于等于IN2

整数比较指令的使用方法与标准触点类似。它可位于任何可放置标准触点的位置。可根据用户选择的比较类型比较IN1和IN2。

如果比较结果为TRUE，则此函数的RLO为“1”。如果以串联方式使用该框，则使用“与”运算将其链接至整个梯级程序段的RLO；如果以并联方式使用该框，则使用“或”运

算将其链接至整个梯级程序段的 RLO。

例如，用比较指令编写一个数据筛选器，如果 MW10 的数在 10 和 100 之间，则指示灯 Q5.0 亮。程序如图 6-43 所示。

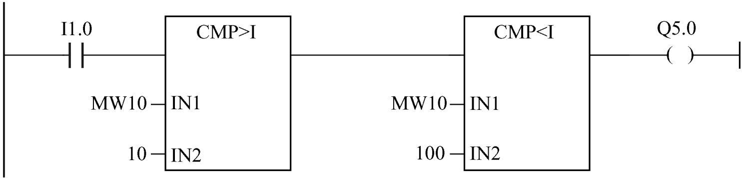


图 6-43　程序图

当 I1.0 为“1”时，判断 MW10 中的数据是否在 10 和 100 之间，若是则 Q5.0 为“1”，如不是则 Q5.0 为“0”。

【例 6-11】 比较存储字 MW10 和输入字 MW20 中整数的大小。如果两个整数相等，则输出 Q 4.0 为 1；若 MW10 中的数大，则输出 Q 4.1 为 1；若 MW20 中的数大，则输出 Q 4.2 为 1。

```
L   MW 10        //第一个待比较的数装入累加器 1
L   MW20         //第二个待比较的数装入累加器 1,第一个数被装入累加器 2
= =I
=Q 4.0           // 若(MW 10)=(MW20),则 Q 4.0 为 1,否则为 0
>I
=Q 4.1           // 若(MW 10)>(MW20),则 Q 4.1 为 1,否则为 0
<I
=Q 4.2           // 若(MWl0)<(MW20),则 Q 4. 2 为 1,否则为 0
```

（2）双整数比较指令

说明见表 6-17。

表 6-17　双整数比较指令

LAD 方块	STL 指令	方块上部的符号	比 较 类 型
CMP==D IN1 IN2	= = D	= =	IN1 等于 IN2
	< > D	< >	IN1 不等于 IN2
	> D	>	IN1 大于 IN2
	< D	<	IN1 小于 IN2
	> = D	> =	IN1 大于等于 IN2
	< = D	< =	IN1 小于等于 IN2

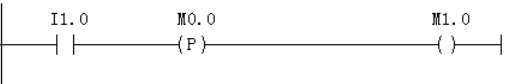
双整数比较指令的使用方法与标准触点类似。它可位于任何可放置标准触点的位置。可根据用户选择的比较类型比较 IN1 和 IN2。

【例 6-12】 用比较指令编写一个求三个数中最大的数的程序。

程序解析：将三个数分别放在 MD20、MD24、MD28 中，在 PLCSIM、组态软件或触摸屏中输入三个数到 MD20、MD24、MD28 中，当 I1.0 为“1”时，程序做相关运算。得出结果后，将三者中的最大数放到 MD36 中。梯形图程序如图 6-44 所示。

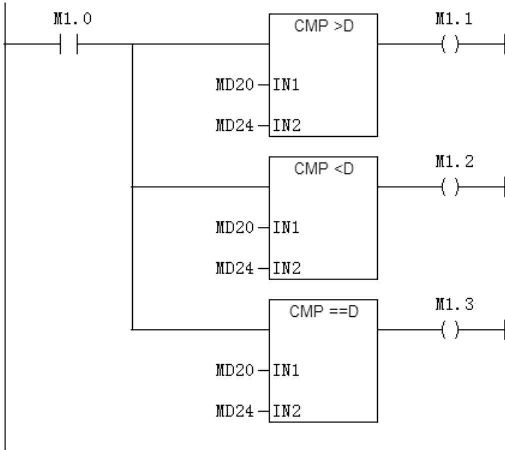
程序段 1：标题：

注释：



程序段 2：标题：

注释：



程序段 3：标题：

注释：

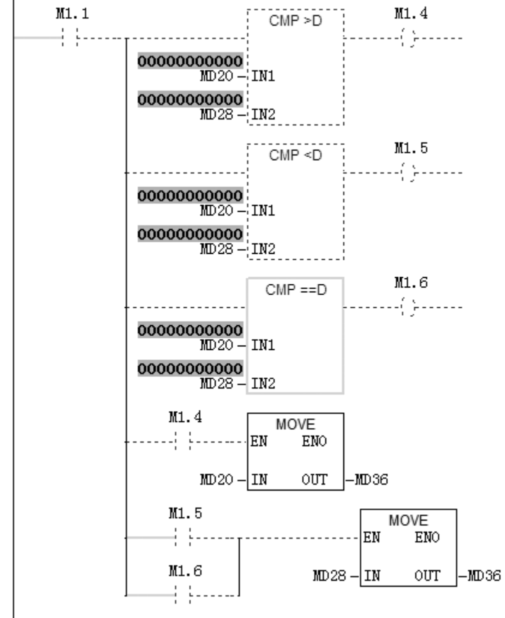


图 6–44　数据比较梯形图

程序段 4：标题：

注释：

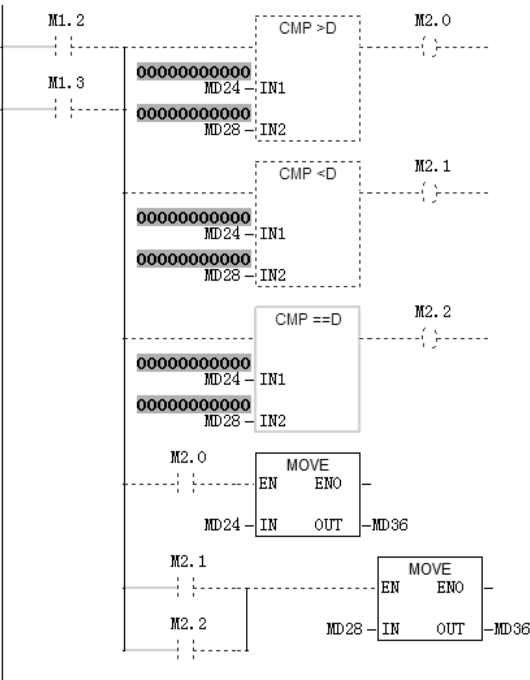


图 6-44　数据比较梯形图（续）

（3）实数比较指令

说明见表 6-18。

表 6-18　实数比较指令

LAD 方块	STL 指令	方块上部的符号	比 较 类 型
CMP==R IN1 IN2	==R	==	IN1 等于 IN2
	<>R	<>	IN1 不等于 IN2
	>R	>	IN1 大于 IN2
	<R	<	IN1 小于 IN2
	>=R	>=	IN1 大于等于 IN2
	<=R	<=	IN1 小于等于 IN2

实数比较指令的使用方法类似标准触点。它可位于任何可放置标准触点的位置。可根据用户选择的比较类型比较 IN1 和 IN2。

【例 6-13】 试用“计数器”“比较器”指令设计。要求按钮 I0.0 闭合 10 次之后，输出 Q4.0；按钮 I0.0 闭合 20 次之后，输出 Q4.1；按钮 I0.0 闭合 30 次之后，计数器及所有输出自动复位。手动复位按钮为 I0.1，计数器类型自己考虑。相应的程序如图 6-45 所示。

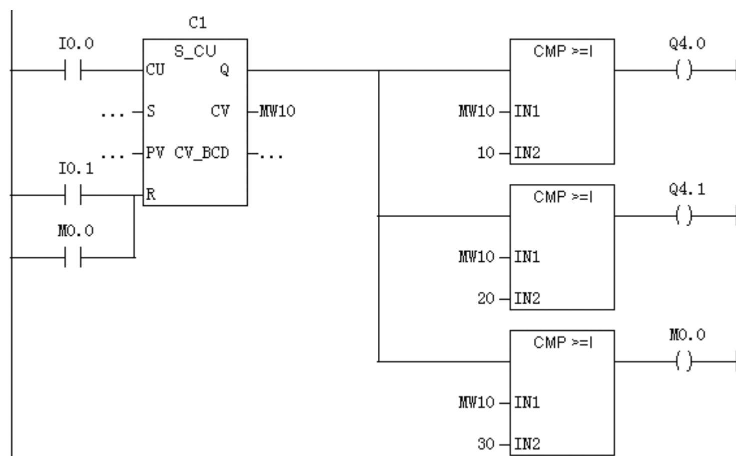


图 6-45　程序梯形图

6.4.3　数据转换指令

在 STEP 7 中数据转换指令的功能是将数据整数、双整数和实数三者之间按照各自指令的规则相互转换。转换指令首先将源数据按照规定的格式读入累加器，然后在累加器中对数据进行类型转换，最后再将转换的结果传送到目的地址。能够实现的转换操作有：BCD 码和整数及长整数间的转换，实数和长整数间的转换，数的取反、取负、字节扩展等。

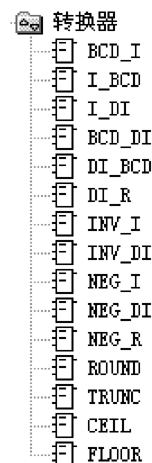


图 6-46　STEP 7 中数据转换指令

在 STEP 7 中，整数和长整数是以补码形式表示的。BCD 码数值有两种：一种是字（16 位）格式的 BCD 码数，其数值范围从 -999 ~ +999；另一种是双字（32 位）格式的 BCD 码数，范围从 -9999999 ~ +9999999。STEP 7 中数据转换指令如图 6-46 所示，各个指令的功能及存储区等信息将在下面讲到。

（1）BCD 和整数间的转换

BCD 和整数之间的转换见表 6-19。

表 6-19　BCD 和整数间的转换

梯形图、STL 指令	参　　数	数 据 类 型	存　储　区	说　　明
BCD_I（EN　END；IN　OUT） BTI 将 3 位 BCD 码转换为 16 位整数	EN	BOOL	I、Q、M、D、L	使能输入
	ENO	BOOL	Q、M、D、L	使能输出
	IN	WORD	I、Q、M、D、L	BCD 码
	OUT	INT	Q、M、D、L	整数

（续）

梯形图、STL 指令	参　　数	数 据 类 型	存　储　区	说　　明
I_BCD EN　END IN　OUT ITB 将 16 位整数转换为 3 位 BCD 码	EN	BOOL	I、Q、M、D、L	使能输入
	ENO	BOOL	Q、M、D、L	使能输出
	IN	INT	I、Q、M、D、L	整数
	OUT	WORD	Q、M、D、L	BCD 码
I_DINT EN　END IN　OUT ITD 将 16 位整数转换为 32 位整数	EN	BOOL	I、Q、M、D、L	使能输入
	ENO	BOOL	Q、M、D、L	使能输出
	IN	INT	I、Q、M、D、L	要转换的值
	OUT	DINT	Q、M、D、L	转换的结果值
BCD_DI EN　END IN　OUT BTD 将 7 位 BCD 码数转换为 32 位整数	EN	BOOL	I、Q、M、D、L	使能输入
	ENO	BOOL	Q、M、D、L	使能输出
	IN	DWORD	I、Q、M、D、L	BCD 码
	OUT	DINT	Q、M、D、L	由 BCD 码转换成的双整数
DI_BCD EN　END IN　OUT DTB 将 32 位整数转换为 7 位 BCD 码数	EN	BOOL	I、Q、M、D、L	使能输入
	ENO	BOOL	Q、M、D、L	使能输出
	IN	DINT	I、Q、M、D、L	双整数
	OUT	DWORD	Q、M、D、L	BCD 码的结果
DI_R EN　END IN　OUT DTR 将 32 位整数转换为 32 位实数	EN	BOOL	I、Q、M、D、L	使能输入
	ENO	BOOL	Q、M、D、L	使能输出
	IN	DINT	I、Q、M、D、L	要转换的值
	OUT	REAL	Q、M、D、L	转换的结果值

注意：数据转换指令中没有整数到实数的直接转换，如果要实现转换，可以通过双整数中间过渡来实现。即 I -> DI -> REAL，先实现从整数到双整数的转换，再实现从双整数到实数的转换，最后完成带有实数的运算程序，如图 6-47 所示。



图 6-47　整数到实数的转换梯形图

（2）实数和长整数间的转换

实数和长整数间的转换见表 6-20。

表 6-20　实数和长整数间的转换

梯形图、STL 指令	参　　数	数 据 类 型	存　储　区	说　　明
ROUND（EN ENO / IN OUT） RND 将实数化整为最接近的整数	EN	BOOL	I、Q、M、D、L	使能输入
	ENO	BOOL	Q、M、D、L	使能输出
	IN	REAL	I、Q、M、D、L	要舍入的值
	OUT	DINT	Q、M、D、L	舍入后的结果
TRUNC（EN ENO / IN OUT） TRUNC 取实数的整数部分（截尾取整）	EN	BOOL	I、Q、M、D、L	使能输入
	ENO	BOOL	Q、M、D、L	使能输出
	IN	REAL	I、Q、M、D、L	要取整的值
	OUT	DINT	Q、M、D、L	IN 的整数部分
CEIL（EN ENO / IN OUT） RND + 将实数化整为大于或等于该实数的最小整数	EN	BOOL	I、Q、M、D、L	使能输入
	ENO	BOOL	Q、M、D、L	使能输出
	IN	REAL	I、Q、M、D、L	要取整的值
	OUT	DINT	Q、M、D、L	上取整后的结果
CEIL（EN ENO / IN OUT） RND − 将实数化整为小于或等于该实数的最大整数	EN	BOOL	I、Q、M、D、L	使能输入
	ENO	BOOL	Q、M、D、L	使能输出
	IN	REAL	I、Q、M、D、L	要取整的值
	OUT	DINT	Q、M、D、L	下取整后的结果

① ROUND（取整为长整型）

将参数 IN 的内容以浮点数读取，并将其转换为长整型（32 位）。结果为最接近的整数（“取整到最接近值”）。如果浮点数介于两个整数之间，则返回偶数。结果由参数 OUT 输出。如果产生溢出，ENO 的状态为“0”。

② TRUNC（截断长整型）

将参数 IN 的内容以浮点数读取，并将其转换为长整型（32 位）。（“向零取整模式”）

的长整型结果由参数 OUT 输出。如果产生溢出，ENO 的状态为“0”。

③ CEIL（上取整）

将参数 IN 的内容以浮点数读取，并将其转换为长整型（32 位）。结果为大于该浮点数的最小整数。如果产生溢出，ENO 的状态为“0”。

④ FLOOR（下取整）

将参数 IN 的内容以浮点数读取，并将其转换为长整型（32 位）。结果为小于该浮点数的最大整数部分。如果产生溢出，ENO 的状态为“0”。

（3）数的取反、取负

数的取反、取负见表 6-21。

表 6-21　数的取反、取负

梯形图、STL 指令	参　数	数据类型	存　储　区	说　明
INV_I（EN ENO / IN OUT） INVI 对 16 位整数求反码	EN	BOOL	I、Q、M、D、L	使能输入
	ENO	BOOL	Q、M、D、L	使能输出
	IN	INT	I、Q、M、D、L	输入值
	OUT	INT	Q、M、D、L	整数的二进制反码，即各位二进制数由 0 变为 1，由 1 变为 0
INV_DI（EN ENO / IN OUT） INVD 对 32 位整数求反码	EN	BOOL	I、Q、M、D、L	使能输入
	ENO	BOOL	Q、M、D、L	使能输出
	IN	DINT	I、Q、M、D、L	输入值
	OUT	DINT	Q、M、D、L	双整数的二进制反码
NEG_I（EN ENO / IN OUT） NEGI 对 16 位整数求补码（取反码再加 1），相当于乘 -1	EN	BOOL	I、Q、M、D、L	使能输入
	ENO	BOOL	Q、M、D、L	使能输出
	IN	INT	I、Q、M、D、L	输入值
	OUT	INT	Q、M、D、L	整数的二进制补码
NEG_DI（EN ENO / IN OUT） NEGD 对 32 位整数求补码	EN	BOOL	I、Q、M、D、L	使能输入
	ENO	BOOL	Q、M、D、L	使能输出
	IN	DINT	I、Q、M、D、L	输入值
	OUT	DINT	Q、M、D、L	双整数的二进制补码
NEG_R（EN ENO / IN OUT） NEGR 对 32 位实数的符号位求反码	EN	BOOL	I、Q、M、D、L	使能输入
	ENO	BOOL	Q、M、D、L	使能输出
	IN	REAL	I、Q、M、D、L	输入值
	OUT	REAL	Q、M、D、L	对输入值求反的结果

① INVI（对整数求反码）

读取 IN 参数的内容，并使用十六进制掩码 W#16#FFFF 执行布尔“异或”运算。此指令将每一位变成相反状态。ENO 始终与 EN 的信号状态相同。

② INVD（对长整数求反码）

读取 IN 参数的内容，并使用十六进制掩码 W#16#FFFF FFFF 执行布尔“异或”运算。此指令将每一位转换为相反状态。ENO 始终与 EN 的信号状态相同。

③ NEGI（对整数求补码）

读取 IN 参数的内容并执行求二进制补码指令，将整数逐位取反后再加 1。二进制补码指令等同于乘以（－1），相当于求一个数的相反数（例如：从正值变为负值）。ENO 始终与 EN 的信号状态相同，以下情况例外：如果 EN 的信号状态＝1 并产生溢出，则 ENO 的信号状态＝0。

④ NEGD（对长整数求补码）

读取参数 IN 的内容并执行二进制补码指令。二进制补码指令等同于乘以（－1）后改变符号（例如：从正值变为负值）。ENO 始终与 EN 的信号状态相同，以下情况例外：如果 EN 的信号状态＝1 并产生溢出，则 ENO 的信号状态＝0。

⑤ NEGR（取反浮点）

读取参数 IN 的内容并改变符号。指令等同于乘以（－1）后改变符号（例如：从正值变为负值）。ENO 始终与 EN 的信号状态相同。

6.4.4　移位指令

移位指令可以将累加器 1 的低字或整个累加器的内容进行左移或右移一定的位数。二进制数左移一位相当于将原数值乘以 2。右移一位相当于将原数值除以 2。

参数 N 表示移位的次数。移出的空位根据不同的指令由 0 或符号位的状态填充。最后移出的位的状态同时被装入到状态字的 CC1 位，CC0 和 OV 位被复位。在 STEP 7 中移位指令如图 6-48 所示。

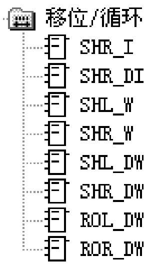


图 6-48　移位指令

（1）无符号移位指令

无符号移位指令的格式见表 6-22。

表 6-22　无符号移位指令

LAD 方块	STL 指令	LAD 方块	STL 指令
SHL_W（EN, ENO, IN, OUT, N）	SLW 将 IN 中的字逐位左移， 空出位填以 0	SHL_DW（EN, ENO, IN, OUT, N）	SLD 将 IN 中的双字逐位左移， 空出位填以 0
SHR_W（EN, ENO, IN, OUT, N）	SRW 将 IN 中的字逐位右移， 空出位填以 0	SHR_W（EN, ENO, IN, OUT, N）	SRD 将 IN 中的双字逐位右移， 空出位填以 0

左移字（SLW）和右移字（SRW）的工作过程如图 6-49 所示。

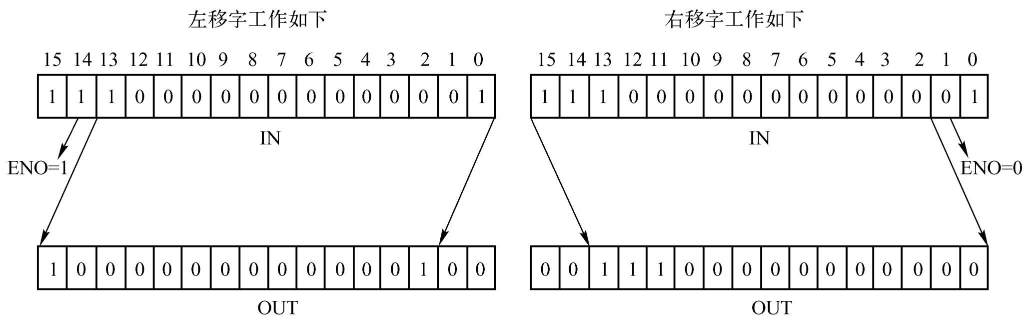


图 6-49　字移位工作过程

【**例 6-14**】将 MW20 中的数左移 7 位。如图 6-50 所示。

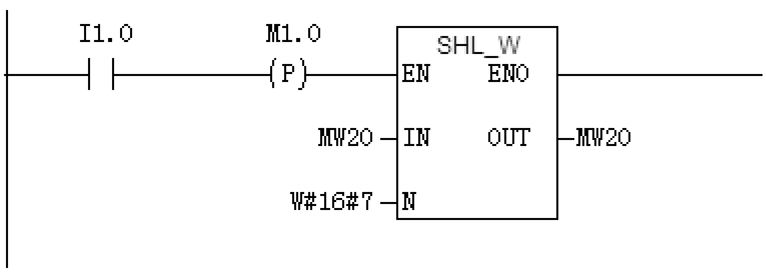


图 6-50　左移梯形图

移位前 MW20 中的数为 B#1100_1111_0011_0000，移位后 MW20 中的数为 B#1001_1000_0000_0000。

（2）有符号移位指令

有符号移位指令的格式见表 6-23。

表 6-23　有符号移位指令

LAD 方块	STL 指令	参　　数	数 据 类 型	存　储　区	说　　明
SHR_I EN　ENO IN　OUT N	SSI 将 IN 中的字逐位右移，空出位填以符号位	EN	BOOL	I、Q、M、D、L	使能输入
		ENO	BOOL	Q、M、D、L	使能输出
		IN	WORD	I、Q、M、D、L	要移位值
		N	WORD	Q、M、D、L	要移位数
		OUT	WORD	I、Q、M、D、L	操作结果
SHR_DI EN　ENO IN　OUT N	SSD 将 IN 中的双字逐位右移，空出位填以符号位	EN	BOOL	I、Q、M、D、L	使能输入
		ENO	BOOL	Q、M、D、L	使能输出
		IN	DINT	I、Q、M、D、L	要移的值
		N	WORD	I、Q、M、D、L	要移位数
		OUT	DINT	Q、M、D、L	操作结果

说明：STL 指令格式：SSI　< number >，其工作原理如下。

① 当使能输入端 EN = 1 时，执行整数右移指令。将来自输入端 IN 的 16 位整数右移 *N* 位后，由 OUT 端输出。

② *N* 端输入要移位的次数，如果 *N* 大于 16，则其作用与 *N* = 16 相同。

③ 移出的空位由符号位的状态填充，如果是正数，以 0 填充，如果是负数，以 1 填充。

④ 如果 *N* 不等于 0，则执行该指令后，CC0 和 OV 位总是等于 0。

⑤ ENO = EN。如图 6-51 所示。

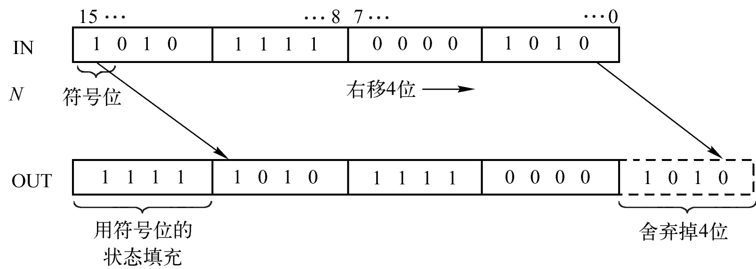


图 6-51　有符号数右移过程

【例 6-15】 将 MW10 中的数右移 5 位。如图 6-52 所示。

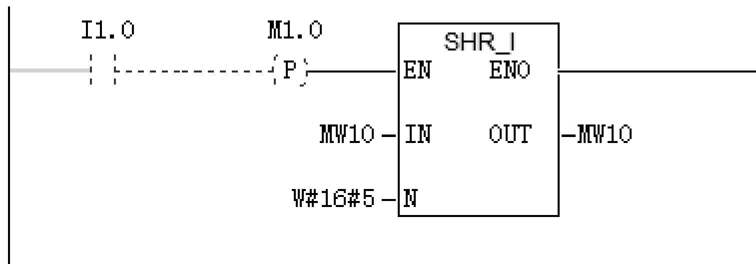


图 6-52　右移梯形图

移位前 MW10 中的数为 B#1111_0011_1111_1100，移位后 MW10 中的数为 B#1111_1111_1001_1111。若移位前 MW10 中的数为 B#0111_0000_0101_1100，移位后 MW10 中的数为 B#0000_0011_1000_0010。

（3）循环移位指令

循环移位指令的格式见表 6-24。

表 6-24　循环移位指令

LAD 方块	STL 指令	参数	数据类型	存　储　区	说　　明
ROL_DW EN　ENO IN　OUT N	RLD 将 IN 中的双字逐位左移，空出位填以移出的位	EN	BOOL	I、Q、M、D、L	使能输入
		ENO	BOOL	Q、M、D、L	使能输出
		IN	WORD	I、Q、M、D、L	要移位值
		N	WORD	Q、M、D、L	要移位数
		OUT	WORD	I、Q、M、D、L	操作结果

（续）

LAD 方块	STL 指令	参数	数据类型	存　储　区	说　　明
ROR_DW EN ENO IN OUT N	RRD 将 IN 中的双字逐位右移，空出位填以移出的位	EN	BOOL	I、Q、M、D、L	使能输入
		ENO	BOOL	Q、M、D、L	使能输出
		IN	DINT	I、Q、M、D、L	要移的值
		N	WORD	I、Q、M、D、L	要移位数
		OUT	DINT	Q、M、D、L	操作结果

说明：STL 指令格式：RLD　< number >

① 当使能输入端 EN = 1 时，执行双字左循环指令。将来自输入端 IN 的 32 位双字左循环 N 位后，由 OUT 端输出。

② N 端输入要移位的次数。

③ 如果 N 不等于 0，则执行该指令后，CC0 和 OV 位总是等于 0。

④ ENO = EN。如图 6-53 所示。

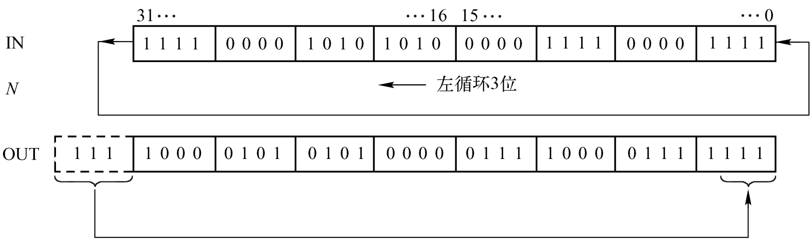


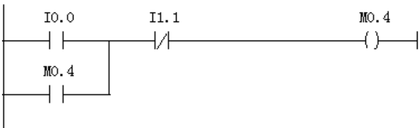
图 6-53　循环左移过程

【例 6-16】移位指令通常需与边缘触发指令配合。走马灯的实现。要求：运用循环移位指令实现 8 个彩灯的循环左移和右移。其中 I0.0 为起停开关，MD20 为设定的初始值，MW12 为移位位数，输出为 Q0.0 ~ Q0.7。

分析：首先建立定时振荡电路，振荡周期为 2.25 s，使得每次定时时间到后，循环移位指令开始移位。在循环移位指令的使用中运用了边缘触发指令，使循环移位在每个定时时间内只移位一次。在程序开始时，必须给循环存储器 MD20 赋初值，比如开始时，只有最低位的彩灯亮（为 1），则初值设定必须为 DW#16#01010101（为了能循环显示，必须设定 MB20、MB21、MB22、MB23 中的值均相同，为 W#16#01，否则，8 位彩灯轮流亮过后，彩灯会有段时间不亮）。梯形图程序如图 6-54 所示。

OB1 : "Main Program Sweep (Cycle)"

程序段 1：启保停电路



程序段 2：启动并赋初值

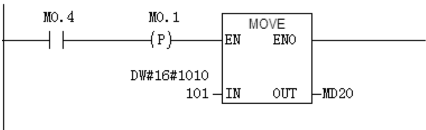
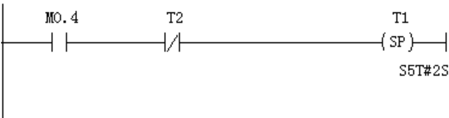


图 6-54　走马灯梯形图

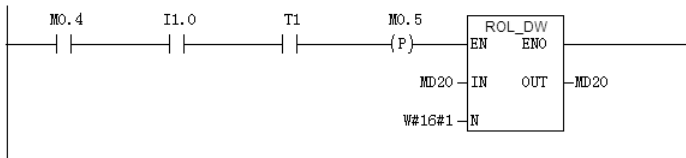
程序段 3：启动定时器，构成振荡电路



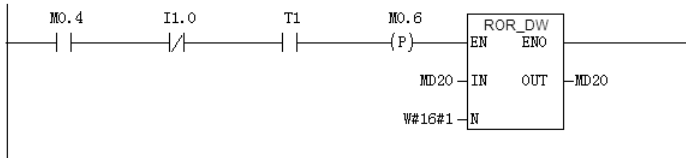
程序段 4：标题：



程序段 5：走马灯左移



程序段 6：走马灯右移



程序段 7：输出控制灯

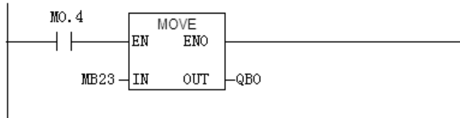


图 6-54　走马灯梯形图（续）

【例 6-17】要求控制 8 盏灯以 1 s 的速度自左向右亮起，并如此反复下去。

方法一：利用 CPU 硬件时钟发生器。方法步骤如下。

① 首先在项目的硬件组态中设置 CPU 的“周期/时钟存储器”，如图 6-55 所示。



图 6-55　时钟存储器硬件设置

② 由表 6-10 可知，MB10 字节的第 5 位能产生周期为 1 s 的方波信号，可以在程序中使用，其梯形图如图 6-56 所示。

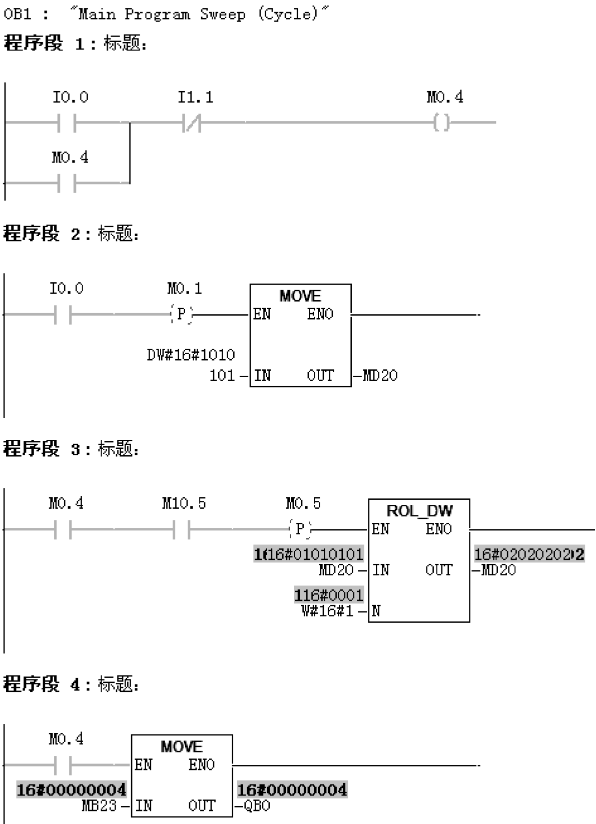


图 6-56　跑马灯梯形图

③ 在 PLCSIM 中进行仿真，其结果如图 6-57 所示。

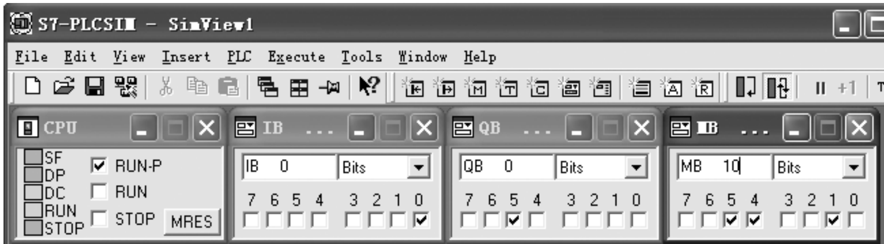


图 6-57　跑马灯仿真监控图

方法二：利用软件定时器产生时间因子，其实现的程序如图 6-58 所示。

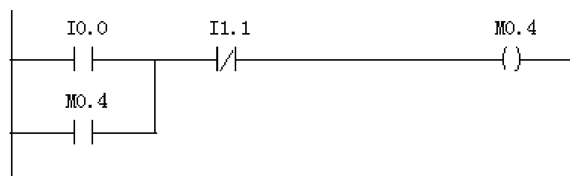
方法三：利用 OB35 的循环中断来产生时间因子。

① S7 提供了多达 9 个的周期性中断 OB（OB30 至 OB38），它们以固定的时间间隔来中断用户程序。首先在项目硬件组态的 CPU 中对 OB35 进行相应的设置。如图 6-59 所示。并对其进行编译下载到 CPU 中。

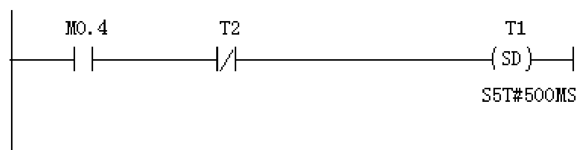
② 在 OB35 中编写周期振荡程序，如图 6-60 所示。

OB1 : "Main Program Sweep (Cycle)"

程序段 1：标题：



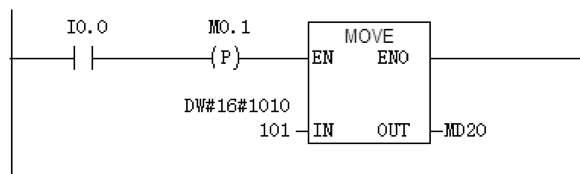
程序段 2：标题：



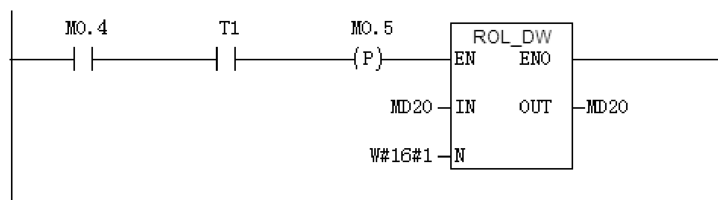
程序段 3：标题：



程序段 4：标题：



程序段 5：标题：



程序段 6：标题：

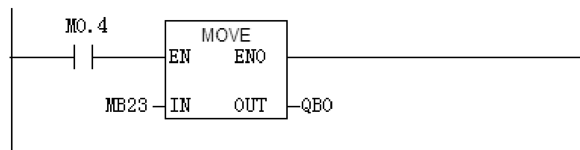


图 6–58　跑马灯梯形图

③ 在 OB1 中编写控制主程序，如图 6–61 所示。主要将移位指令的控制端进行更换即可。



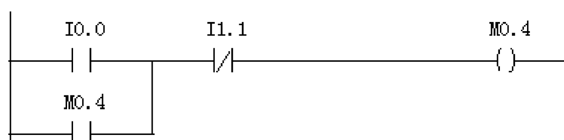
图 6-59　OB35 参数设置



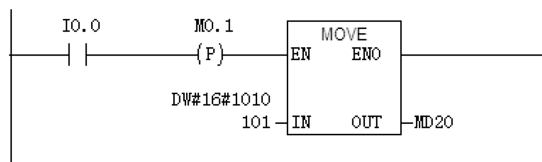
图 6-60　周期振荡梯形图

OB1 :　"Main Program Sweep (Cycle)"

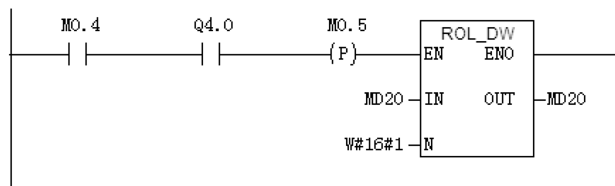
程序段 1：标题:



程序段 2：标题:



程序段 3：标题:



程序段 4：标题:

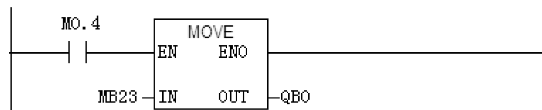


图 6-61　跑马灯梯形图

总结：以上三种方法控制思想一样，主要是用于产生控制移位的时间脉冲方法不一样，

也可以把这些周期脉冲产生的方法应用在其他的程序中，主要在于灵活的使用。

6.5　运算指令

现代 PLC 实际上是一台工业控制计算机，一般都具有很强的运算能力。对 S7－300 PLC，运算指令有两大类，即算术运算指令和逻辑运算指令。

6.5.1　算术运算指令

算术运算十分重要，因为一般的自动控制系统都需要 PID 控制器，其算法的实现离不开基本的算术运算。在 STEP 7 中可以对整数、长整数和实数进行加、减、乘、除算术运算。算术运算指令在累加器 1 和 2 中进行，在累加器 2 中的值作为被减数或被除数。算术运算的结果保存在累加器 1 中，累加器 1 原有的值被运算结果覆盖，累加器 2 中的值保持不变。

CPU 在进行算术运算时，不必考虑 RLO，对 RLO 也不产生影响。学习算术运算指令必须注意算术运算的结果将对状态字的某些位产生影响，这些位是：CC1 和 CC0，OV，OS。在位操作指令和条件跳转指令中，经常要对这些标志位进行判断来决定进行什么操作。

整数与浮点数数学运算指令属于 STEP 7 中的两大类运算指令，它们能够完成大多数的基本运算，比如，加减乘除、三角函数等。

（1）整数数学运算指令

见表 6-25。

表 6-25　数学运算指令

LAD	STL 指令	方块上部的符号	说　明
ADD_I EN　ENO IN1 IN2　OUT	+I	ADD_I	将 IN1 和 IN2 中的 16 位整数相加，结果保存到 OUT 中
	-I	SUB_I	IN1 整数减去 IN2，结果在 OUT 中
	*I	MUL_I	IN1 和 IN2 整数相乘，结果以 32 位整数存到 OUT 中
	/I	DIV_I	将 IN1 中 16 位整数除以 IN2 中的 16 位整数，商保存到 OUT 中
SUB_DI EN　ENO IN1 IN2　OUT	+D	ADD_DI	IN1 和 IN2 的双字相加，存到 OUT
	-D	SUB_DI	IN1 减去 IN2，结果保存到 OUT 中
	*D	MUL_DI	将 IN1 和 IN2 中的 32 位整数相乘，结果保存到 OUT 中
	/D	DIV_DI	将 IN1 中 32 位整数除以 IN2 中的 32 位整数，商保存到 OUT 中
	MOD	MOD	将 IN1 中 32 位整数除以 IN2 中的 32 位整数，余数保存到 OUT 中

整数数学运算指令比较简单，不再给出梯形图程序，其语句表指令使用如下所示。

```
L   MW0     // 将 MW 0 中的值装入累加器 1 低字
L   MW2     // 将 MW 2 中的值装入累加器 1 低字,累加器 1 低字中的原值移入累加器 2 低字
+I          // 将累加器 1 低字和累加器 2 中的低字相加
T   MW10    // 将运算结果送到 MW 10
```

（2）浮点数数学运算指令

S7－300 系列 CPU 可以处理符合 IEEE 标准的 32 位浮点数。可以完成 32 位浮点数的加、

减、乘、除运算，以及取绝对值、平方、开平方、指数、对数、三角函数、反三角函数等指令。浮点数数学运算指令见表 6-26。

表 6-26　浮点数数学运算指令

STL 指令	梯　形　图	说　　明
+R	ADD_R	将累加器 1、2 中的 32 位浮点数相加，32 位结果保存在累加器 1 中
-R	SUB_R	用累加器 2 中的 32 位浮点数减去累加器 1 中的浮点数，结果保存在累加器 1 中
*R	MUL_R	将累加器 1、2 中的 32 位浮点数相乘，32 位乘积保存在累加器 1 中
/R	DIV_R	用累加器 2 中的 32 位浮点数除以累加器 1 中的浮点数，32 位商保存在累加器 1 中
ABS	ABS	对累加器 1 中的 32 位浮点数取绝对值
SQR	SQR	求累加器 1 中的 32 位浮点数的平方值
SQRT	SQRT	求累加器 1 中的 32 位浮点数的开平方根
EXP	EXP	求累加器 1 中的 32 位浮点数以 e 为底的指数
LN	LN	求累加器 1 中的 32 位浮点数的自然对数
SIN	SIN	求累加器 1 中的 32 位浮点数的正弦值
COS	COS	求累加器 1 中的 32 位浮点数的余弦值
TAN	TAN	求累加器 1 中的 32 位浮点数的正切值
ASIN	ASIN	求累加器 1 中的 32 位浮点数的反正弦值
ACOS	ACOS	求累加器 1 中的 32 位浮点数的反余弦值
ATAN	ATAN	求累加器 1 中的 32 位浮点数的反正切值

【例 6-18】 运用算术运算指令完成下面的方程式运算，其梯形逻辑图如图 6-62 所示。

$$MW4 = ((IW\ 0 + DBW\ 3) \times 15) / MW\ 0$$

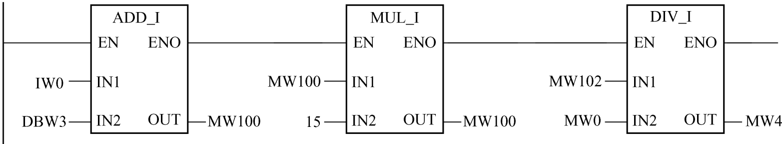


图 6-62　算术运算梯形图

实现例 6-18 相同运算的语句表如下。

```
L     IW0        //将输入字 IW0 的内容装入累加器 1
L     DBW3       //将 DBW3 的内容装入累加器 1,累加器 1 原内容装入累加器 2
 +I              //累加器 2 与累加器 1 相加,结果存在累加器 1 中
L      +15       //将常数 15 装入累加器 1,累加器 1 原内容(和)装入累加器 2
 *I              //累加器 2 与累加器 1 相乘,结果为长整数放在累加器 1 中
L     MW0        //将存储双字 MD0 的内容装入累加器 1,累加器 1 原内容装入累加器 2
/I               //累加器 2 除累加器 1,结果的整数部分存在累加器 1 中
T     MW4        //将运算结果传送至存储双字 MD4
```

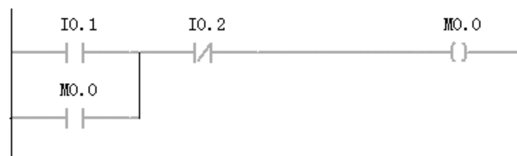
完成相同运算功能的梯形图程序和语句表各有优缺点：梯形图程序直观易读；语句表简洁，而且使用中间结果存储器较少。

【例 6-19】 求算式 1 +2 +3 + … +100 的值。

方法一：线性化常规编程实现其算式的梯形图如图 6-63 所示。

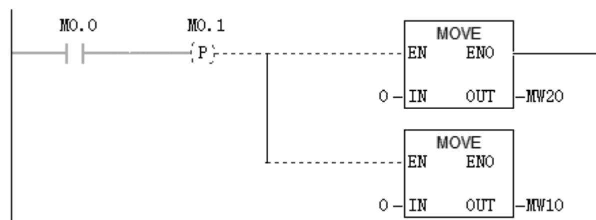
程序段 1：标题：

注释：



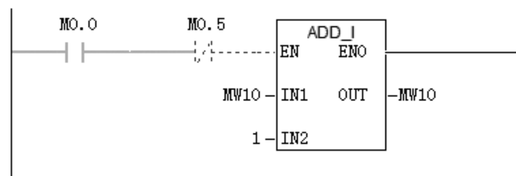
程序段 2：标题：

注释：



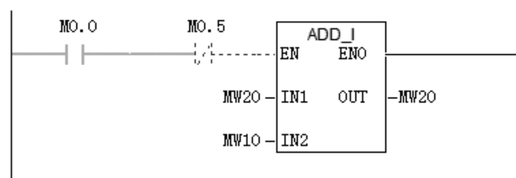
程序段 3：标题：

注释：



程序段 4：标题：

注释：



程序段 5：标题：

注释：

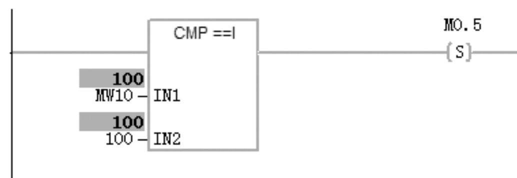


图 6-63　求和梯形图

在 PLCSIM 中进行仿真结果如图 6-64 所示。

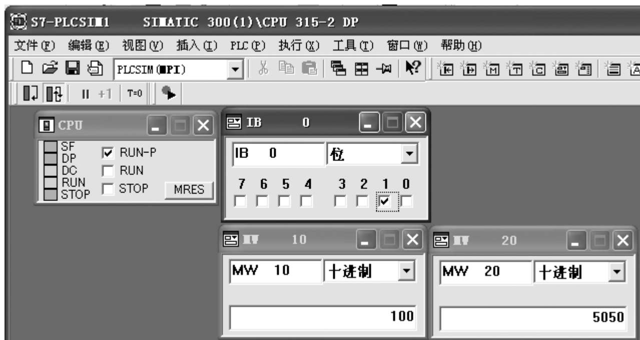
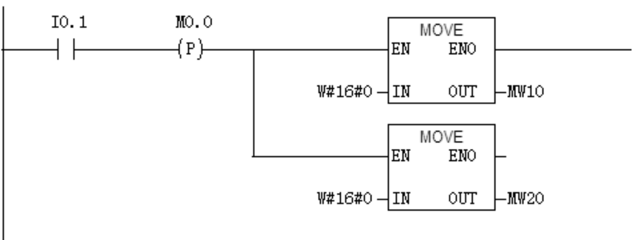
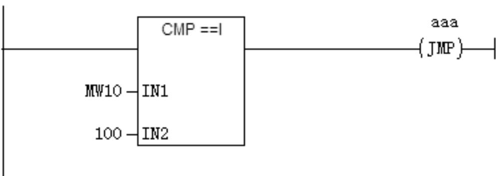
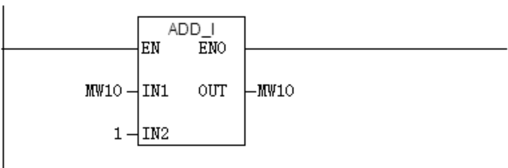


图 6-64　求和仿真结果

方法二：利用逻辑控制指令，其实现的梯形图如图 6-65 所示。







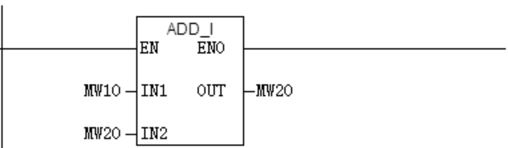


图 6-65　求和梯形图

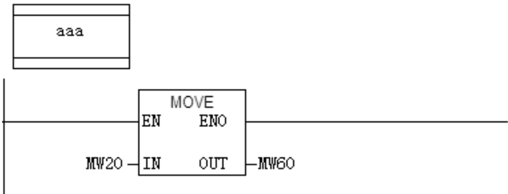


图 6-65　求和梯形图（续）

方法三：利用 LOOP 指令实现其算式的语句表如图 6-66 所示。

L　L#0	0	1	0
T　MW　20	0	1	0
L　L#100	0	1	64
NEXT: T　MW　10	0	1	64
+I	0	1	64
T　MW　20	0	1	64
L　MW　10	0	1	64
LOOP　NEXT			

图 6-66　利用 LOOP 指令实现算式

【例 6-20】设计一个简易计算器，具有两个整数加减乘除的功能。

（1）I/O 分配

如图 6-67 所示。

	状态	符号	地址	数据类型
1		开始	I　0.0	BOOL
2		清零	I　0.1	BOOL
3		加	I　0.2	BOOL
4		减	I　0.3	BOOL
5		乘	I　0.4	BOOL
6		除	I　0.5	BOOL
7		数据1	MD　10	DWORD
8		数据2	MD　20	DWORD
9		结果	MD　36	DWORD
10				

图 6-67　简易计算器 I/O 分配

（2）梯形图程序

如图 6-68 所示。

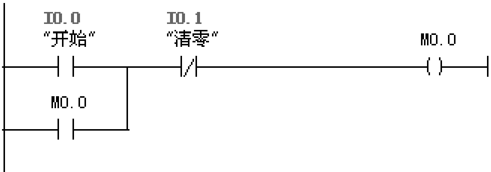
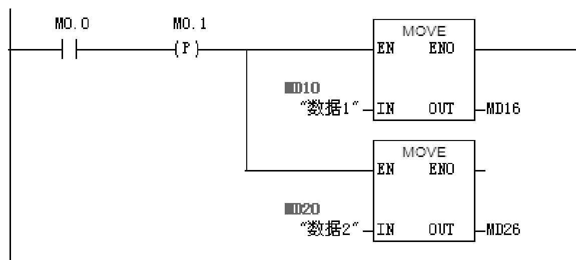


图 6-68　简易计算器梯形图

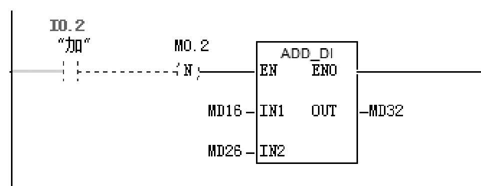
程序段 2：标题：

注释：



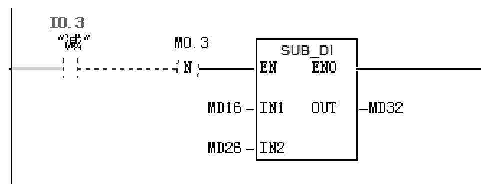
程序段 3：标题：

注释：



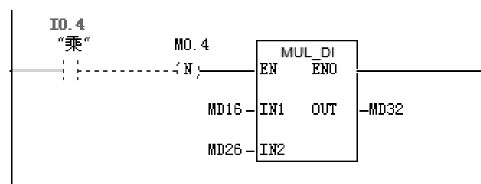
程序段 4：标题：

注释：



程序段 5：标题：

注释：



程序段 6：标题：

注释：

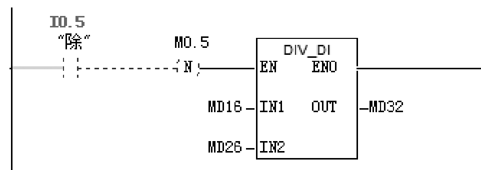


图 6–68　简易计算器梯形图（续）

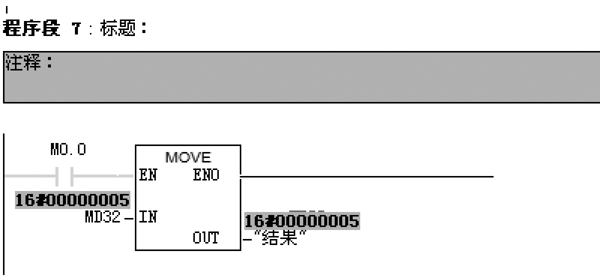


图 6-68　简易计算器梯形图（续）

【例 6-21】 编写程序实现 1 +3 +5 + ⋯ +99 的算式实现，要求用语句表编程实现，并将最后结果存储到 MW20 中。

```
        L       0
        T       MW      20
        L       99
JIA:    T       MW      10
        L       MW      20
        +I
        T       MW      20
        L       1
        L       MW      10
        ==I
        JC      XX
        DEC     1
XX:LOOP         JIA
```

【例 6-22】 编写程序实现 2 +4 +6 + ⋯ +100 的算式实现，要求用语句表编程实现，并将最后结果存储到 MW20 中。

```
        L       0
        T       MW      20
        L       100
JIA:    T       MW      10
        L       MW      20
        +I
        T       MW      20
        L       MW      10
        DEC     1
LOOP    JIA
```

6.5.2　字逻辑运算指令

字逻辑指令将两个字（16 位）或两个双字（32 位）逐位进行逻辑运算。两个数中的一个在累加器 1 中，另一个可以在累加器 2 中或在指令中以立即数（常数）的方式给出，字

逻辑运算指令的逻辑运算结果放在累加器 1 低字中，双字逻辑运算结果存储在累加器 1 中，累加器 2 的内容保持不变。

逻辑运算结果影响状态字的标志位。如果逻辑运算的结果为 0，则 CCl 位被复位为 0。如果逻辑运算的结果非 0，则 CCl 被置为 1。

在任何情况下，状态字中的 CC0 和 OV 位都被复位为 0。在 STEP 7 中字逻辑运算指令包括单字与运算（WAND_W ）、单字或运算（WOR_W）、单字异或运算（WXOR_W）、双字与运算（WAND_DW）、双字或运算（WOR_DW）、双字异或运算（WXOR_DW）。字逻辑运算指令见表 6–27。

表 6–27　字逻辑运算指令表

LAD 方块	STL 指令	方块上部的符号	功 能 说 明
WAND_W EN ENO IN1 OUT IN2	AW	WAND_W	16 位字逻辑与指令
	OW	WOR_W	16 位字逻辑或指令
	XOW	WXOR_W	16 位字逻辑异或指令
WAND_DW EN ENO IN1 OUT IN2	AD	WAND_DW	32 位字逻辑与指令
	OD	WOR_DW	32 位字逻辑或指令
	XOD	WXOR_DW	32 位字逻辑异或指令

字逻辑运算指令使用比较简单，按位进行相应的运算即可。比如对于 16 位字逻辑与指令 AW，当使能输入端 EN 输入为“1”时，将来自 IN1 和 IN2 端的两个 16 位二进制数据逐位相与，结果由 OUT 端输出。使能输出端 ENO = 使能输入端 EN。运算如下：

IN1 = 0101010101010101；

IN2 = 0000000000001111；

OUT = 0000000000000101；

字逻辑运算指令梯形图使用如图 6–69 所示，在 PLCSIM 中监控如图 6–70 所示。

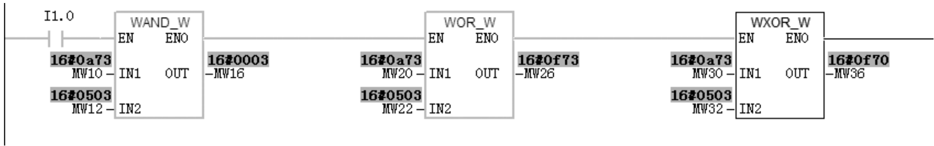


图 6–69　字逻辑运算梯形图

6. 5. 3　累加器指令

累加器是 PLC 中的一个重要元件，是程序执行中经常用到的寄存器之一，在数据处理和数字运算中都使用了累加器。直接对累加器进行操作，有助于处理程序中的一些问题，表 6–28 列出了对累加器进行直接操作的主要指令。

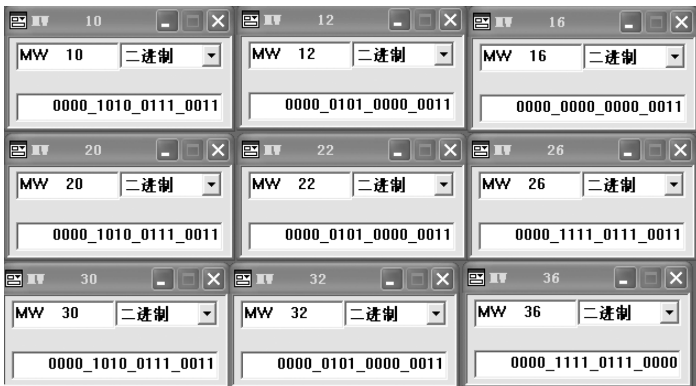


图 6-70　PLCSIM 监控图

表 6-28　累加器直接操作指令

STL 指令	名　　称	说　　明
TAK	互换	累加器 1 和累加器 2 的内容互换
PUSH	压入	把累加器 1 的内容移入累加器 2，累加器 2 原内容被丢掉
POP	弹出	把累加器 2 的内容移入累加器 1，累加器 1 原内容被丢掉
INC	增加	把累加器 1 低字的低字节内容加上指令中给出的常数，常数范围：0 ~255；指令的执行是无条件的，结果不影响状态字
DEC	减少	把累加器 1 低字的低字节内容减去指令中给出的常数，常数范围：0 ~255；指令的执行是无条件的，结果不影响状态字
CAW	反转	交换累加器 1 低字中的字节顺序
CAD	交换	交换累加器 1 中的字节顺序

对表 6-28 中的指令说明如下。

① CAW 是将累加器 1 低字的高位字节和低位字节交换，高字不变。交换顺序见表 6-29。

表 6-29　CAW 操作指令

	ACCU1_H－H	ACCU1_H－L	ACCU1_L－H	ACCU1_L－L
CAW 指令执行前	数据 A	数据 B	数据 C	数据 D
CAW 指令执行后	数据 A	数据 B	数据 D	数据 C

② CAD 是累加器 1 中的 4 个字节进行整字节交换。交换顺序见表 6-30。

表 6-30　CAW 操作指令

	ACCU1_H－H	ACCU1_H－L	ACCU1_L－H	ACCU1_L－L
CAD 指令执行前	数据 A	数据 B	数据 C	数据 D
CAD 指令执行后	数据 D	数据 C	数据 B	数据 A

③ INC 指令在循环中起比较重要的作用。

【例 6-23】 该例说明了在有条件触发的程序中 INC 指令是如何工作的。本例以存储字节 MB10 作为循环次数计数暂存器，用 INC 指令修正循环次数，循环体中的程序连续执行 5 次。

```
LOOP:L     MB10          //循环体开始,装载存储字节至累加器 1
```

```
INC    1                  //循环计数器加 1
T      MB10               //保存循环次数
L      B#16#5
<=I
JC     LOOP               //如果循环次数小于等于 5 次,则继续循环
L      1                  //循环体外的操作,为下次循环做好准备
T      MB10
```

6.6　控制指令

控制指令主要指控制程序执行的顺序和控制程序结构的有关指令。包括逻辑控制指令、程序控制指令、主控继电器指令和块调用指令。

6.6.1　逻辑控制指令

（1）逻辑控制指令

逻辑控制指令是指逻辑块内的跳转和循环指令，这些指令改变了程序原有的线性逻辑流，使程序转移到另外的程序地址，重新开始扫描。转移到的新地址用目标地址标号（简称标号）表示，在一个逻辑块内的标号是唯一的，不能重复，在不同的逻辑块内，标号可以相同。标号最多为 4 个字符，第一个字符必须是字母，其余字符可以是字母或数字（例如，CAS1）。每个—（JMP）或—（JMPN）都还必须有与之对应的跳转标号（LABEL）。

在 STL 指令中，标号或必须紧接冒号，后接目标指令。在 LAD 指令中，标号必须在一个网络的开始，可在梯形图程序编辑器上，从编程元件浏览器中选择 LABLE（标号），在出现的空方块中填上标号。

PLC 的程序一般是各条语句按从上到下顺序逐条执行的，这种执行方式称为线性扫描。但有时因某种原因，如因控制或运算的需要，需中断程序的线性扫描，跳过程序中的某一部分再继续按线性扫描方式向下执行，则要用到跳转指令。跳转指令分为无条件跳转指令和条件跳转指令两种。在 STEP 7 中的跳转指令如图 6-71 所示。

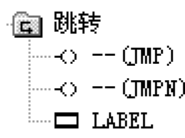


图 6-71　跳转指令

（2）无条件跳转指令（STL 形式的为 JU，Jump Unconditional）

无条件跳转指令（JU）将无条件中断正常的程序逻辑流，使程序跳转到目标处继续执行。如图 6-72 所示。

（3）条件跳转指令

条件跳转指令先要判断跳转的条件是否满足，若满足，程序调到指定的目标标号处继续执行；若不满足，程序不跳转，顺序执行。条件跳转指令见表 6-31。

表 6-31　条件跳转指令表

STL 指令	说　　明
JC	当 RLO =1 时跳转
JCN	当 RLO =0 时跳转

（续）

STL 指令	说　　明
JCB	当 RLO＝1 且 BR＝1 时跳转，将 RLO 保存在 BR 中
JNB	当 RLO＝0 且 BR＝0 时跳转，将 RLO 保存在 BR 中
JBI	当 BR＝1 时跳转，指令执行时，OR、FC 清 0，STA 置 1
JNBI	当 BR＝0 时跳转，指令执行时，OR、FC 清 0，STA 置 1
JO	当 OV＝1 时跳转
JOS	当 OS＝1 时跳转，指令执行时，OS 清 0
JZ	累加器 1 中的计算结果为零跳转
JN	累加器 1 中的计算结果为非零跳转
JP	累加器 1 中的计算结果为正跳转
JM	累加器 1 中的计算结果为负跳转
JMZ	累加器 1 中的计算结果小于等于零（非正）跳转
JPZ	累加器 1 中的计算结果大于等于零（非负）跳转
JUO	实数溢出跳转

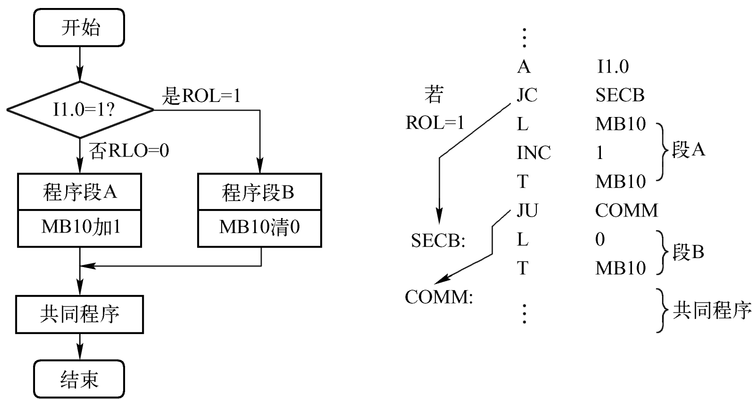


图 6-72　无条件跳转指令实例图

（4）循环指令

使用循环指令（LOOP）可以多次重复执行特定的程序段，重复执行的次数存在累加器 1 中，即以累加器 1 为循环计数器。LOOP 指令执行时，将累加器 1 低字中的值减 1，如果不为 0，则回到循环体开始处继续循环过程，否则执行 LOOP 指令后面的指令。循环体是指循环标号和 LOOP 指令间的程序段。

LOOP 指令以 WORD 数据类型作为循环计数器来判断，由于循环次数不能是负值，所以程序应保证作为循环计数器的累加器 1 中的值为正整数（数值范围：0～32767），或者是字型数据（数值范围：W#16#0000～W#16#FFFF）。存储区为 I、Q、M、D、L。在图 6-73 中，考虑到循环体（程序段 A）中可能用到累加器 1，特设置了循环计数暂存器 MB10。

【例 6-24】 用循环指令求 10 的阶乘。

用语句表实现如下：

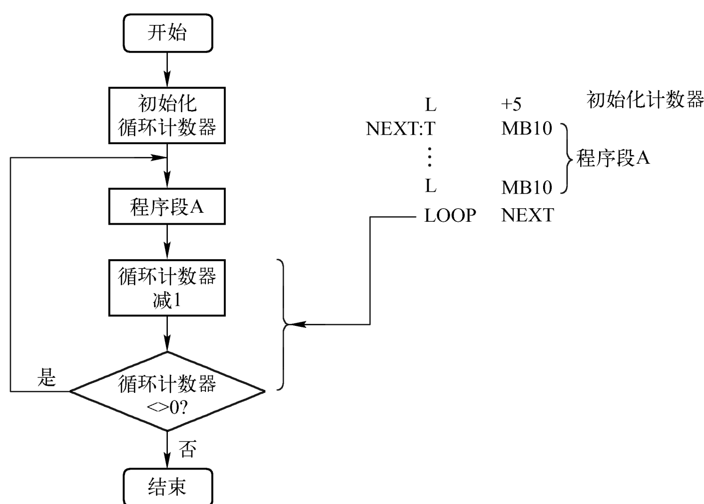


图 6-73　循环指令实例图

```
        L      L#1          //初始化累乘器为 1
        T      MD     20
        L      10           //循环次数装入累加器中
JIEC:   T      MW     10    //循环次数存入 MW10 中
        L      MD     20
        *D                  //MD20 与 MW10 中内容相乘
        T      MD     20    //乘积结果存入 MD20 中
        L      MW     10
        LOOP JIEC           //如果累加器中的循环次数减 1 后大于零,则跳转到标号处继续执行
```

（5）梯形图逻辑控制指令

梯形逻辑控制指令只有两条，可用于无条件跳转或条件跳转控制。由于无条件跳转时对应 STL 指令 JU，因此不影响状态字；由于在梯形图中目的标号只能在梯形网络的开始处，因此条件跳转指令会影响到状态字。

a. —(JMP)—无条件跳转

当左侧电源轨道与指令间没有其他梯形图元素时执行的是绝对跳转。每一个—（JMP）都还必须有与之对应的目标（LABEL）。跳转指令和标号间的所有指令都不予执行。如图 6-74 所示。始终执行跳转，并忽略跳转指令和跳转标号间的指令。

b. —(JMP)—条件跳转

—（JMP）（为 1 时在块内跳转）当其前面逻辑运算的 RLO 为“1”时执行的是条件跳转。每一个—（JMP）都还必须有与之对应的目标（LABEL）。跳转指令和标号间的所有指令都不予执行。如图 6-75 所示。

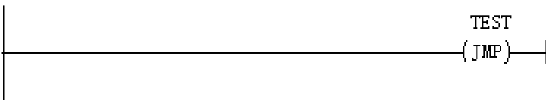
如果 I0. 1 =“1”，则执行跳转到标号 TEST。由于此跳转的存在，即使 I1. 0 处有逻辑“1”，也不会使 Q5. 1 =“1”。

c. —(JMPN)若”否”则跳转

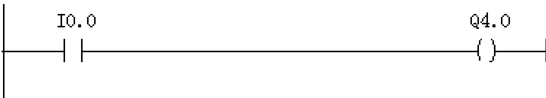
—（JMPN）（若“否”则跳转）相当于在 RLO 为“0”时执行的“转到标号”功能。每一个 —（JMPN）都还必须有与之对应的目标（LABEL）。跳转指令和标号间的所有指令

OB1 : "Main Program Sweep (Cycle)"

程序段 1：标题：



程序段 2：标题：



程序段 3：标题：

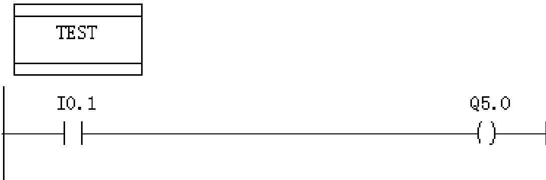
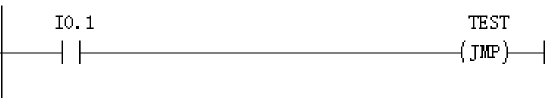


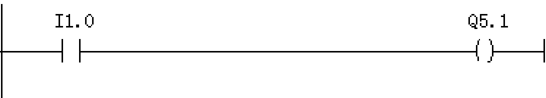
图 6-74　无条件跳转例图

OB1 : "Main Program Sweep (Cycle)"

程序段 1：标题：



程序段 2：标题：



程序段 3：标题：



图 6-75　条件跳转例图

都不予执行。

（6）状态位触点指令

状态位指令是针对状态字的各个位进行操作的指令，通过状态位可以判断 CPU 运算中溢出、异常、进位、比较结果等状态。在 STEP 7 中还有大量的状态位触点指令可以使用，如图 6-76 所示。

图 6-77 所示的程序中，利用状态位指令以及跳转指令实现了当网络 1 的程序运算结果

有溢出时，跳过网络 3 和网络 4 直接执行网络 5 程序，若运算结果无溢出则执行网络 3 和网络 4 的程序并跳过网络 5 直接执行网络 6 程序的功能。

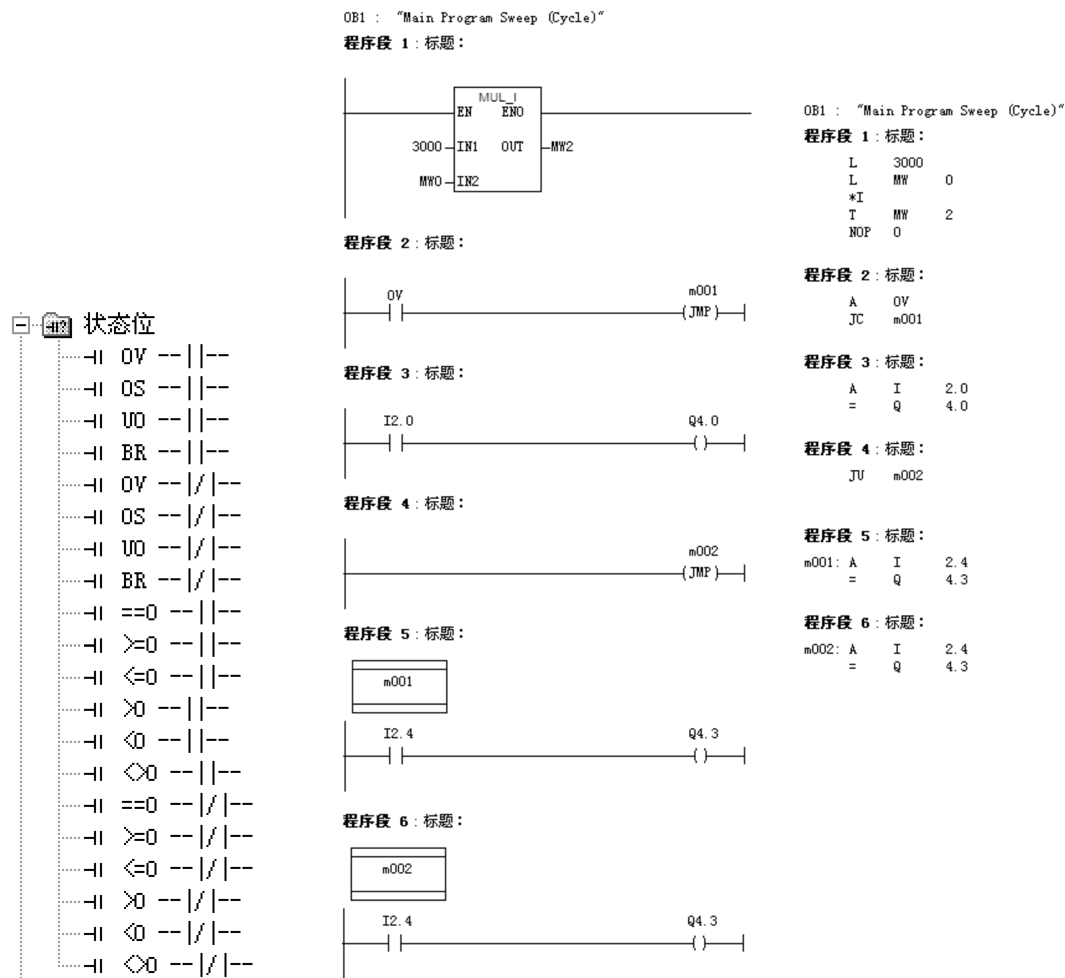


图 6-76　状态位触点指令　　　　图 6-77　状态位指令使用举例

6.6.2　程序控制指令

程序控制指令是指功能块（FB、FC、SFB、SFC）调用指令和逻辑块（OB、FB、FC）结束指令。调用块或结束块可以是有条件的或是无条件的。STEP 7 中的功能块实质上就是子程序。在 STEP 7 中如图 6-78 所示。

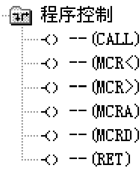


图 6-78　程序控制指令

（1）STL 程序控制指令（表 6-32）

表 6-32　程序控制语句表

指　　令	说　　明
CALL	该指令在程序中无条件执行，调用 FB、FC、SFB、SFC
UC	该指令在程序中无条件调用功能块（一般是 FC 或 SFC），但不能传递参数

（续）

指　　令	说　　明
CC	RLO＝1，调用功能块（一般是 FC），但不能传递参数
BE	块结束指令
BEU	该指令无条件结束当前块的扫描，将控制返回给调用块
BEC	若 RLO＝1，则结束当前块的扫描，将控制返回给调用块 若 RLO＝0，则将 RLO 置 1，程序继续在当前块内扫描

（2）梯形图程序控制指令（见表 6-33）

表 6-33　梯形图程序控制指令

LAD 指令	参　　数		数 据 类 型	存　储　区	说　　明
< FC/SFC no. > ——(CALL)	FC/SFC	no.	BLOCK_FC	—	no. 为被调用的不带参数的 FC 或 SFC 号数
<DB no.> FB no. EN　ENO	方块上部符号	参数			
	FB no.	DB no.	BLOCK_DB	—	调用 FB 时背景数据块号
	FC no.	Block no.	BLOCK_FB/BLOCK_FC	—	被调用的功能块号
	SFB no.	EN	BOOL	I、Q、M、D、L	允许输入
	SFC no.	ENO	BOOL	Q、M、D、L	允许输出
——(RET)	—		—	—	块结束

说明：

① ——（Call）（调用来自线圈的 FC/SFC）

——（Call）（不带参数调用 FC 或 SFC）用于调用没有传递参数的功能（FC）或系统功能（SFC）。只有在 CALL 线圈上 RLO 为“1”时，才执行调用。

② ——（Call）（RET）返回

RET（返回）用于有条件地退出块。对于该输出，要求在前面使用一个逻辑运算。

6.6.3　主控继电器指令

主控继电器（Master Control Relay，MCR）是一种美国梯形图逻辑主控开关，用于控制信号流（电流路径）的通断，它可以控制 MCR 区内的指令是否被正常的执行，相当于一个用于接通和断开“能流”的主令开关。MCR 指令用得并不多，S7－200 PLC 中没有 MCR 指令。其指令表 6-34。

表 6-34　STEP 7 中与主控继电器相关的指令

STL 指令	LAD 指令	说　　明
MCRA	——（MCRA）	激活 MCR 区，表明一个 MCR 区域的开始
MCRD	——（MCRD）	表明一个按 MCR 方式操作区域的结束
MCR（	——（MCR <）	主控继电器，并产生一条母线（子母线）
）MCR	——（MCRA >）	恢复 RLO，结束子母线，返回主母线

说明：

① ——（MCRA）主控制继电器激活

——（MCRA）（激活主控制继电器）激活主控制继电器功能。在该命令后，可以使用

下列命令编程 MCR 区域：

——（MCR < ）

——（MCR > ）

② ——（MCRD）主控制继电器取消激活

——（MCRD）（取消激活主控制继电器）取消激活 MCR 功能。在该命令后，不能编程 MCR 区域。

③ ——（MCR < ）主控制继电器打开

——（MCR < ）（打开主控制继电器区域）在 MCR 堆栈中保存 RLO。MCR 嵌套堆栈为 LIFO（后入先出）堆栈，且只能有 8 个堆栈条目（嵌套级别）。当堆栈已满时，——（MCR < ）功能产生一个 MCR 堆栈故障（MCRF）。——（MCR > ）主控制继电器关闭

——（MCR > ）（关闭最后打开的 MCR 区域）从 MCR 堆栈中删除一个 RLO 条目。其使用如图 6-79 所示。

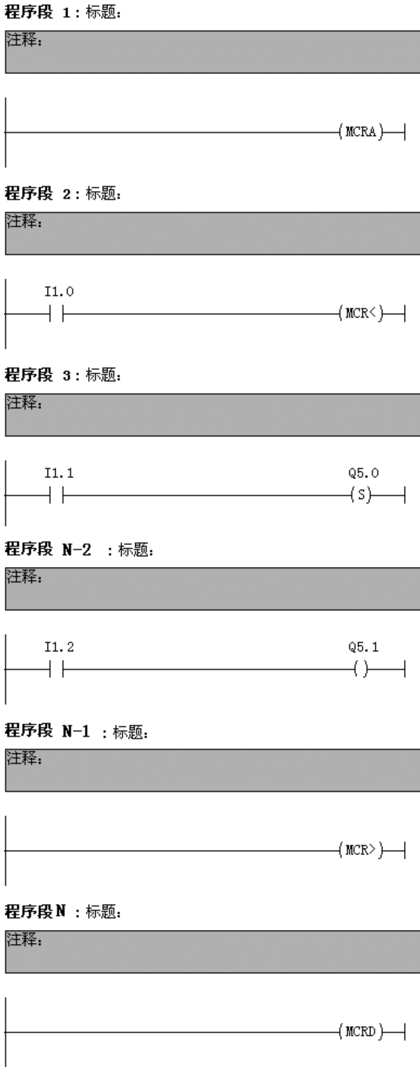


图 6-79　主控制继电器例图

MCRA 梯级激活 MCR 功能。MCR < 和 MCR >（输出 Q5.0、Q5.1）之间的梯级按如下执行。

I1.0 =“1”（MCR 打开）：当 I1.1 为逻辑“1”时，将 Q5.0 设置成“1”，或当 I1.1 为“0”时，Q5.0 保持不变，并将 I1.2 的逻辑状态分配给 Q5.1。

I1.0 =“0”（MCR 关闭）：无论 I1.1 的逻辑状态如何，Q5.0 保持不变，无论 I1.2 的逻辑状态如何，Q5.1 为“0”。

在下一个梯级中，指令—（MCRD）取消激活 MCR。这表示不能再使用指令对—（MCR <）和—（MCR >）编程更多的 MCR 区域。

注意：MCRA 和 MCRD 指令必须成对使用。编程在 MCRA 和 MCRD 之间的指令根据 MCR 位的信号状态执行。编程在 MCRA 和 MCRD 程序段之外的指令与 MCR 位的信号状态无关。

6.6.4 数据块指令

（1）数据块指令

① 打开数据块指令格式：OPN <data block>

说明：打开一个数据块作为共享数据块（DB）或者作为背景数据块（DI）。可以同时打开一个共享数据块和一个背景数据块，打开的共享数据块和背景数据块的编号分别存储在 DB 寄存器和 DI 寄存器中。打开新的数据块后，原来打开的数据块自动关闭。调用一个功能块时，它所关联的背景数据块被自动打开。如果该功能块调用了其他的逻辑块，调用结束后返回该功能块，原来打开的背景数据块不再有效，必须重新打开它。其使用如下所示。

```
OPN   DB 10    //打开数据块 DB 10 作为共享数据块
LDB   W35      // 将 DB 10 的数据字 W35 装入到累加器 1 的低字。
T     M W22    //将累加器 1 的低字传输到 M W22。
OPN   DI 20    //打开数据块 DI 20 作为背景数据块。
L     DI B12   //将 DI 20 的数据字节 B12 装入到累加器 1 的低字。
T     DB B37   //将累加器 1 的低字传输到 DB 10 的字节 37 中。
```

② 交换共享数据块和背景数据块，其指令格式：CDB

说明：交换共享数据块和背景数据块，共享数据块变成背景数据块，背景数据块变成共享数据块。

③ 装共享数据块的长度到累加器 1，指令格式：L　DBLG

④ 装共享数据块的编号到累加器 1，指令格式：L　DBNO

⑤ 装背景数据块的长度到累加器 1，指令格式：L　DILG

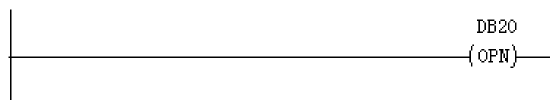
⑥ 装背景数据块的编号到累加器 1，指令格式：L　DINO

⑦ 在梯形图中，与数据块操作有关的只有一条无条件打开共享数据块或背景数据块的指令，即—（OPN）打开数据块：DB 或 DI。—（OPN）函数是一种对数据块的无条件调用。将数据块的编号传送到 DB 或 DI 寄存器中。

【例 6-25】 图 6-80 是数据块打开和数据访问的例子。首先打开数据块 DB20，触点地址（DBX1.0）引用包含在 DB20 中的当前数据记录的数据字节 1 的第 0 位，并将此位的信号状态分配给输出 Q5.0。

OB1 : "Main Program Sweep (Cycle)"

程序段 1：标题：



程序段 2：标题：

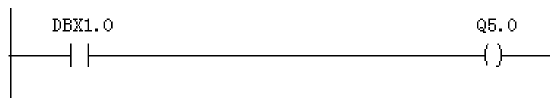


图 6-80　数据块指令使用举例

【例 6-26】 调用数据块 DB20，当数据块长度超过 30 Byte 时，程序转移到标号为 ABC 处，调用功能 FC15。用 STL 编写的程序如下所示。

```
OPN      DB20
L        DBLG
L        +30
>=I
JC       ABC
A        I0.1
BEU
ABC:     CALL     FC15
```

（2）显示和空操作指令

显示和空操作指令表 6-35。这些指令是在梯形图转换为语句表指令时自动添加的，无特别的意义。但如果在语句表中将这些指令去掉，语句表指令就不能再自动转换为梯形图。

表 6-35　显示和空操作指令

STL 指令	说　明
BLD	控制编程器显示程序的形式，不影响程序的执行
NOP 0	空操作 0
NOP 1	空操作 1

6.7　库分类及应用

6.7.1　库的分类

库用于存储 SIMATIC S7/M7 中可多次使用的程序部件。这些程序部件可从已有的项目中复制到库中，也可以直接在库中生成。该库与其他项目无关。程序中的库如图 6-81 所示。

如果在 S7 Program 下的库中存储有用户希望多次调用的块，可节省大量的编程时间并提高效率。可以将这些程序块复制到用户程序中所需要的地方。

在 STEP 7 标准软件包中标准程序库如图 6-82 所示，其中标准程序库包含有下面的功能及功能块。



图 6-81　程序中的库

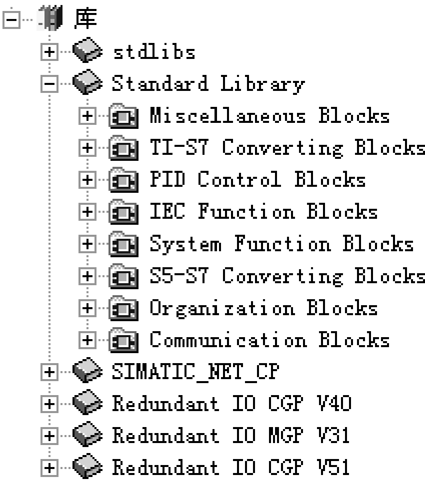


图 6-82　标准程序库

① TI－S7 Converting Blocks（TI－S7 转换块）：通用标准功能，如模拟数值的规范化等。

② PID Control Blocks（PID 控制块）：用于 PID 控制的功能块（FB）。

③ IEC Function Blocks（IEC 功能块）：用于 IEC 功能的块，例如处理时间和日期信息、比较操作、字符串处理以及选择最大值和最小值。

④ System Function Blocks（系统功能块）：包括系统功能块（SFB）和系统功能（SFC）。

⑤ S5－S7 Converting Blocks（S5－S7 转换块）：包含将 S5 程序转换成 S7 程序所需的标准功能块，用于转换 STEP 5 程序的块。

⑥ Organization Blocks（组织块）：标准组织块（OB）。

⑦ Communication Blocks（通信块）：包含使用 S7－300 PROFIBUS CP 时连接分布式 I/O 的块。

当安装其他软件包时，可增加其他库，附加库是在安装可选包时创建的。

6.7.2　库的应用

【例 6-27】 I0.0 为电动机的启/停按钮，Q4.0 和 Q4.1 为两个电动机负载。要求：当按下 I0.0 5 s 后负载 Q4.0 启动，再计时 4 s 后负载 Q4.1 启动；当 I0.0 为“0”时，负载 Q4.0 和 Q4.1 同时停止。

编写程序可以直接调用标准库中的系统功能块 SFB4 TON（通电延时定时器），具体功能见 IEC 定时器，程序如图 6-83 所示。

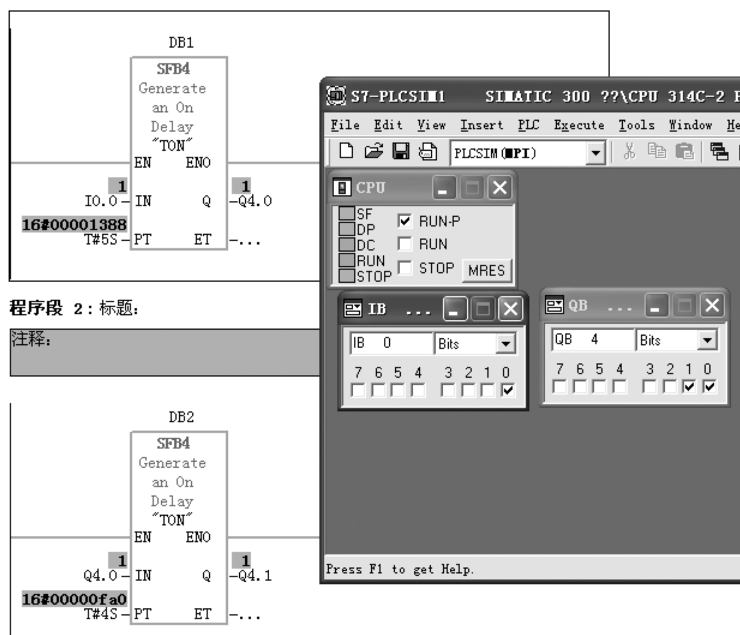


图 6-83　调用库中的系统功能块

6.7.3　库的生成

用户可以将项目中编写的功能或功能块添加进库中，应用于其他项目。如要创建一个库，可按如下操作进行：在 SIMATIC 管理器中，选择菜单命令“文件”→“新建”，弹出提示对话框，选择“库”选项卡，如果希望将新库插入到一个多重项目中，请选择复选框“插入当前的多重项目中”。只有当已在项目窗口中预先选择了多重项目时，才可使用复选框。在对话框中，为新库输入一个名称，在“类型”栏选择“库”，单击“确定”生成一个新的库，如图 6-84 所示。



图 6-84　创建用户库

创建库后出现一个拆分窗口，在窗口的左半部分显示代表库的图标。可将 S7/M7 程序从项目复制到库中，或创建库中的 S7/M7 程序（使用菜单命令“插入”→“程序”→“S7 程序”或→“插入”→“程序”→“M7 程序”）。如图 6-85 所示。

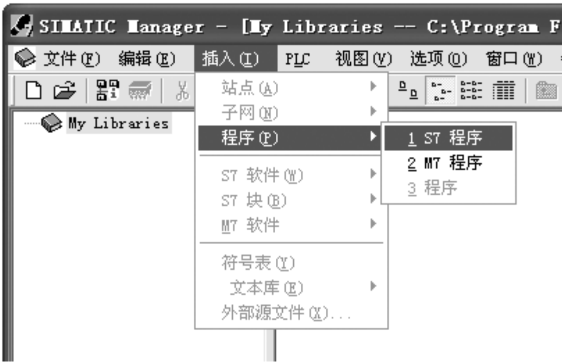


图 6-85　生成库

将编写的功能或功能块可拖放到图 6-84 所生成的“块”中，如图 6-86 所示。即可在编程界面中调用，如图 6-87 所示。还可以对所创建的库进行复制和删除等操作。

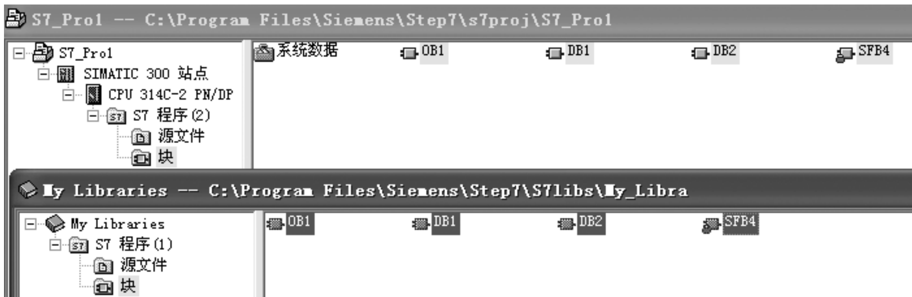


图 6-86　编辑库

【例 6-28】 SAVE 指令将 RLO 的状态存入 BR 寄存器，如果没有存储 BR 位信号，编写的函数或函数块使用 LAD 程序直接调用时，ENO 不能直接输出，函数显示为虚线，如图 6-88所示，FC1 的 ENO 不能输出，M1.0 不能为 1。

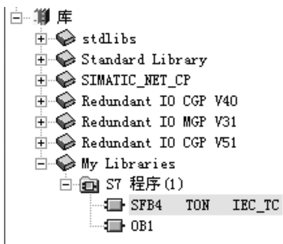


图 6-87　库的调用

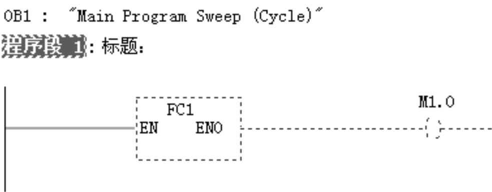


图 6-88　没有将 RLO 状态存入 BR 位调用 FC1 的显示状态

从程序显示上看，FC1 似乎没有调用，实际已经调用，只是显示问题，在调用 FC1 前加入条件（常开或常闭触点）即可，如图 6-89 所示。或者 FC1 的程序结尾使用“—

(SAVE)”指令，可以改变 FC1 调用的显示状态，如图 6-90 和图 6-91 所示。

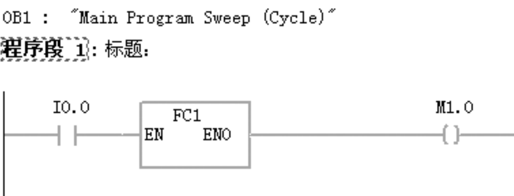


图 6-89　调用 FC1 前加入控制条件的显示状态

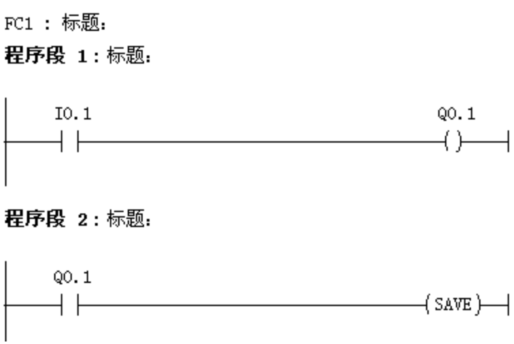


图 6-90　FC1 中添加 SAVE 指令

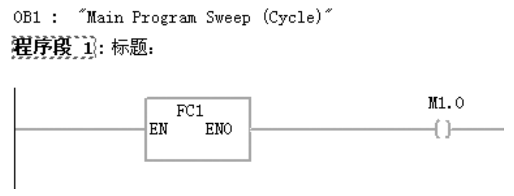


图 6-91　将 RLO 状态存入 BR 位后调用 FC1 的显示状态

【例 6-29】 有一部电动运输小车供 7 个加工点使用，对小车的控制有以下几点要求。

① PLC 上电后，车停在某加工点（下称工位），若没有用车呼叫（下称呼车）时，则各工位的指示灯亮，表示各工位可以呼车。

② 若某工位呼车（按本工位的呼车按钮）时，则各工位的指示灯均灭，表示此后再呼车无效。

③ 停车位呼车则小车不动。当呼车位号大于停车位号时，小车自动向高位行驶，当呼车位号小于停车位号时，小车自动向低位行驶。当小车到达呼车位号时自动停车。

④ 小车到达某工位时停留 30 s 供该工位使用，不立即被其他工位呼走。

⑤ 临时停电再复电，小车不会自行启动。

对于本例的要求，可参照下面的步骤进行程序设计。

① 通过对控制要求的分析得知：每个工位应设一个限位开关和一个呼车按钮。系统要有启动和停止按钮并有指示灯做呼车显示。小车由电机拖动，需正反转控制（各需一个接触器）。各工位的限位开关和呼车按钮如图 6-92 所示。

② 根据分析，选择机型，做出 I/O 分配表，见表 6-36。

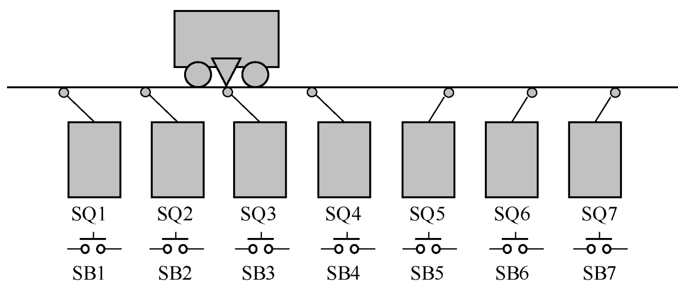


图 6-92　限位开关和呼车按钮布置图

表 6-36　I/O 分配表

输　入				输　出	
信号名称	地址	信号名称	地址	信号名称	地址
启动按钮	I0.0	停止按钮	I1.0	电机正转接触器线圈	Q4.0
限位开关 SQ1	I0.1	呼车按钮 SB1	I1.1		
限位开关 SQ2	I0.2	呼车按钮 SB2	I1.2	电机反转接触器线圈	Q4.1
限位开关 SQ3	I0.3	呼车按钮 SB3	I1.3		
限位开关 SQ4	I0.4	呼车按钮 SB4	I1.4	呼车指示灯	Q4.2
限位开关 SQ5	I0.5	呼车按钮 SB5	I1.5		
限位开关 SQ6	I0.6	呼车按钮 SB6	I1.6		
限位开关 SQ7	I0.7	呼车按钮 SB7	I1.7		

③ 为了分析方便可以先做出系统控制流程图，如图 6-93 所示。

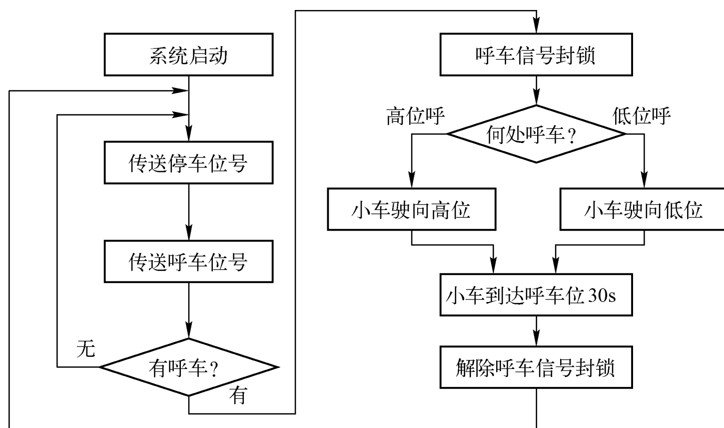


图 6-93　小车 PLC 控制流程图

④ 编写程序。

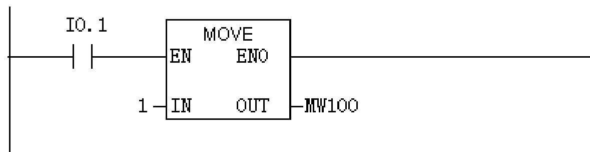
a. 对照本例要求，一般会想到用 MOVE 指令去传送工位号和呼车号．程序如图 6-94 所示。

FC1 : 传送停车工位号

注释：

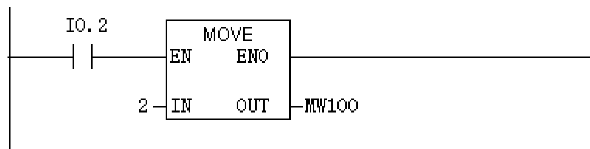
程序段 1：标题：

注释：



程序段 2：标题：

注释：

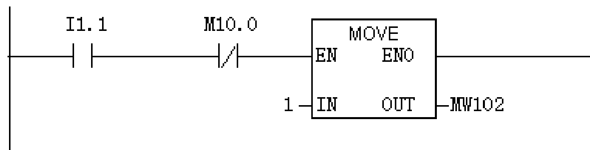


FC4 : 传送呼车工位编号

把呼车车工位号存在MW102中

程序段 1：标题：

注释：



程序段 2：标题：

注释：

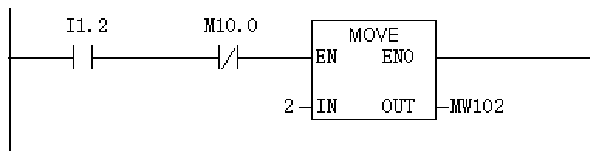


图 6-94 PLC 控制程序

b. 用 CMP 指令去判断小车的行驶方向。程序如图 6-95 所示。

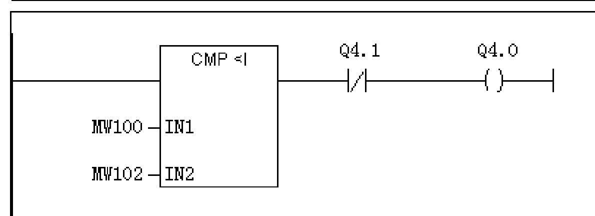
c. 再总体考虑系统启动、呼车指示等功能的实现。程序如图 6-96 所示。

FC3 ： 小车行驶方向判断

注释：

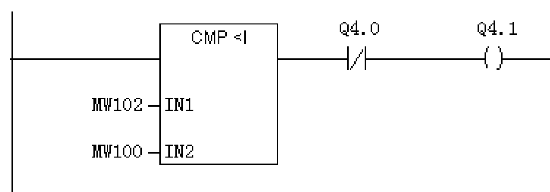
程序段 1：标题：

停车位小于呼车位，小车正转驶向高位



程序段 2：标题：

呼车位小于停车位，小车反转驶向低位



程序段 3：标题：

呼车位等于停车位，小车停留30s

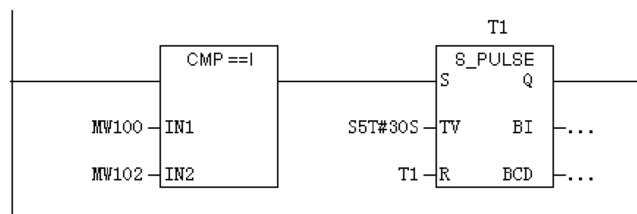


图 6-95　PLC 控制程序

程序段 1：标题：

系统启动

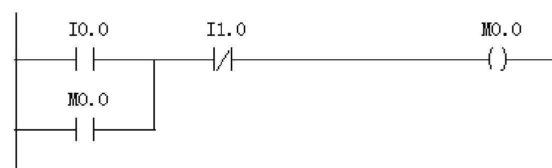
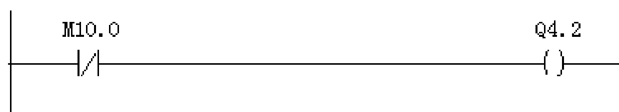


图 6-96　PLC 控制程序

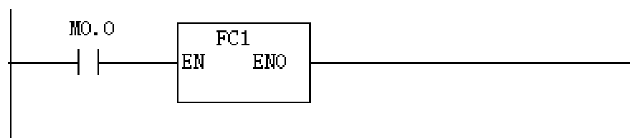
程序段 2：标题：

呼车指示



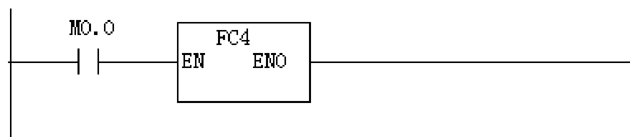
程序段 3：标题：

调用工位传送程序



程序段 4：标题：

调用工位传送程序



程序段 5：标题：

调用小车运动程序

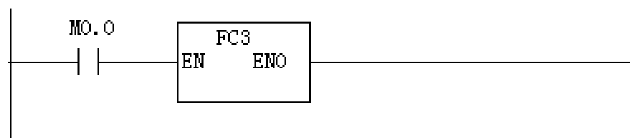


图 6-96　PLC 控制程序（续）

习题

6-1　对触点的边沿检测指令与对 RLO 的边沿检测指令有何区别？

6-2　S7－300 系列 PLC 有几种形式的定时器？脉冲定时器和扩展脉冲定时器有何区别？

6-3　S7－300 系列 PLC 的计数器有几种计数方式？

6-4　依据图 6-97 所示的 LAD 程序以及 I0.0 的时序图，画出 Q4.0 和 Q4.1 的输出波形（I0.1 的状态总为 OFF）。

6-5　设计一个对锅炉鼓风机和引风机控制的梯形图程序。控制要求如下。

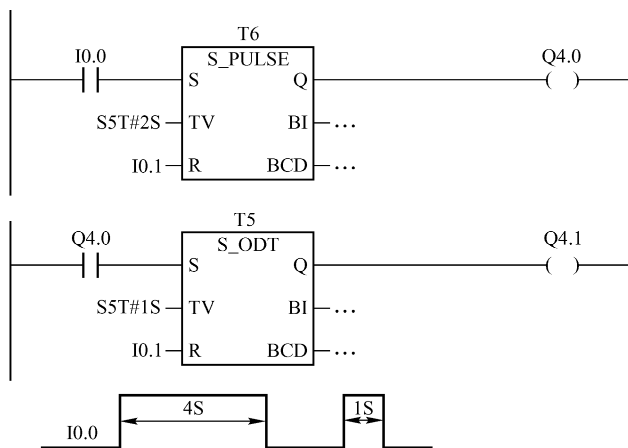


图 6-97　题 6-4 图

（1）开机时首先启动引风机，10 s 后自动启动鼓风机。

（2）停止时立即关断鼓风机，20 s 后自动关断引风机。

6-6　分析图 6-98 中程序功能。（I0.0 为点击按钮，机械上不能自锁，Q0.0 表示一个灯。）

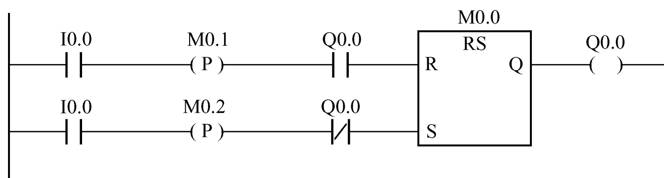


图 6-98　题 6-6 图

6-7　分析下列程序段，请填写执行后相应寄存器的数值。

```
L     DW#16#12345678
T     MD     0
L     MD     0
CAD
T     MD     4
```

请填写：MB4 中的数据是___________；累加器 ACCU1 中的数据是___________；累加器 ACCU2 中的数据是_____________。

6-8　分析下列程序段，请填写执行后相应寄存器的数值。

```
L      -90.7
T     MD        20
RND -
T     MD        24
L     MD        20
TRUNC
T     MD        20
```

请填写：MD24 中的数据是_________；MD20 中的数据是____________。

6-9　编一个 2 s ON，4 s OFF 的占空比可调的脉冲发生器程序。

6-10　用 STEP7 的算术逻辑指令完成算术运算："(235.5 + 125.0) × 13.7 ÷ 7.8 = ?"；试画出其完成运算的梯形图。要求：用 I0.0（外接常开点）启动运算，用 MD20 存储计算结果。

6-11　如图 6-99 所示为某仓库区及显示面板。在两个传送带之间有一个装 100 件物品的仓库，传送带 1 将物品送至临时仓库。传送带 1 靠近仓库区一端的光电传感器（I0.0）确定有多少物品运送至仓库区，传送带 2 将仓库区中的物品运送至货场，传送带 2 靠近仓库区一端的光电传感器（I0.1）确定已有多少物品从库区送至货场。显示面板上有 5 个指示灯（Q12.0 ~ Q12.4）显示仓库区物品的占有程度。编写梯形图来实现其功能。

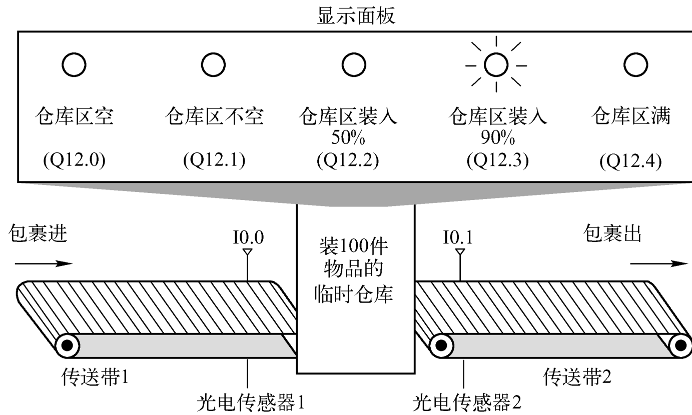


图 6-99　题 6-11 图

6-12　编写 PLC 控制程序，使 Q4.0 输出周期为 5 s，占空比为 20% 的连续脉冲信号。

6-13　用 I0.0 控制 Q0.0 ~ Q0.7 每隔 1 s 顺序置高电平一次，用循环移位和定时器指令实现。

6-14　多级皮带运输机控制。图 6-100 是一个四级传送带系统示意图。整个系统有四台电动机 M1、M2、M3、M4，落料漏斗 Y0 由一阀控制。控制要求如下。

（1）落料漏斗启动后，传送带 M1 应马上启动，经 6 s 后须启动传送带 M2。

（2）传送带 M2 启动后 5 s 后应启动传送带 M3。

（3）传送带 M3 启动后 4 s 后应启动传送带 M4。

（4）落料停止后，应根据所需传送时间的差别，分别隔 6 s、5 s、4 s、3 s 将 4 台电机停车。要求画出简单的 I/O 分配以及 PLC 外围接线图，并编写梯形图实现控制任务。

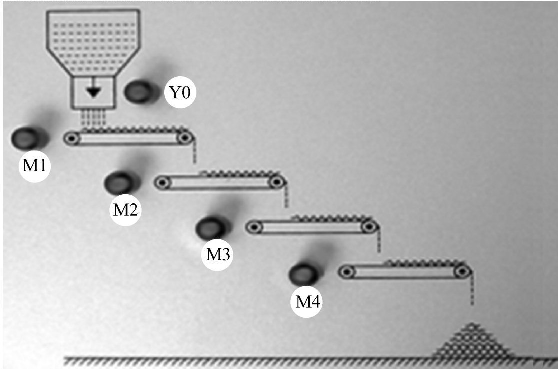


图 6-100　题 6-14 图

6-15　库有哪些分类？如何生成一个新的用户库？怎样从库中调用功能及功能块？

第 7 章　结构化编程

S7-300/400 系列 PLC 采用 STEP 7 编程语言，用文件块的形式来管理用户编写的程序及程序运行所需要的数据，组成结构化的用户程序。这样的组织使得 PLC 程序结构清晰、组织明确，便于修改。

7.1　编程方式和程序结构

7.1.1　编程方式

用户的编程方式主要有线性化编程、分部式编程和结构化编程三种类型。

线性化编程是将整个用户程序放在组织块 OB1 中，在 CPU 循环扫描时执行 OB1 中的全部指令。其特点是线性程序的结构简单，分析起来一目了然。这种结构适用于编写一些规模较小，运行过程比较简单的控制程序。然而某些相同或相近的操作需要多次执行，这样会造成不必要的编程工作。再者，由于程序结构不清晰，会造成管理和调试的不方便。所以在编写大型程序时，应避免线性化编程。

分部式编程是将整个程序按任务分成若干个部分，并分别放置在不同的功能（FC）、功能块（FB）及组织块中，在一个块中可以进一步分解成段。在组织块 OB1 中包含按顺序调用其他块的指令，并控制程序执行。在分部式程序中，既无数据交换，也不存在重复利用的程序代码。功能（FC）和功能块（FB）不传递也不接收参数，分部式程序结构的编程效率比线性程序有所提高，程序测试也较方便，对程序员的要求也不太高。对不太复杂的控制程序可考虑采用这种程序结构。

但是对于复杂的自动控制任务上述两种编程方式则显得捉襟见肘。故而引出第三种编程方式——结构化编程。结构化编程是将过程要求类似或相关的任务归类，在功能或功能块中编程，形成通用解决方案。通过不同的参数调用相同的功能或通过不同的背景数据块调用相同的功能块。其特点是每个块在 OB1 中可能会被多次调用，以完成具有相同过程工艺的不同控制对象，这种结构必须对系统功能进行合理分析、分解和综合，以简化程序设计过程、提高编程的效率。

7.1.2　程序结构

西门子公司 S7-300/400 系列 PLC 采用的是“块式程序结构”，用“块”的形式来管理用户编写的程序及程序运行所需要的数据，组成完整的 PLC 应用程序系统。

（1）用户程序使用的块

在 S7-300 PLC 中，PLC 的程序分为操作系统和用户程序。操作系统处理的是底层的系统级任务，它为 PLC 应用搭建了一个平台，提供了一套用户程序的调用机制，主要用于实现与特定的控制任务无关的功能，比如处理 PLC 的启动、刷新输入/输出过程映像表、调用

用户程序、处理中断和错误、管理存储区和处理通信等。用户程序在这个搭建的平台上完成自己特定的自动化控制任务，而这个程序需要用户自己编写。通常用户程序主要完成暖启动和热启动的初始化工作、处理过程数据（数字信号、模拟信号）、对中断的响应、对异常和错误的处理等工作。

通常用户编写的程序和程序过程中所需要的数据均被 STEP 7 放置在块中，使单个的程序标准化。这样做的好处是：块与块之间的调用结构化、简单化、透明化，易于理解、修改、查错和调试。

用户程序是由块组成的，构成用户程序的逻辑块包括组织块 OB（Organization Block）、功能块 FB（Function Block）、功能 FC（Function）、系统功能块 SFB（System Function Block）、系统功能 SFC（System Function）等。各种块的简单说明见表 7-1。

表 7-1　用户程序中的块

块	简要描述
组织块（OB）	操作系统与用户程序的接口，决定用户程序的结构
系统功能块（SFB）	集成在 CPU 模块中，通过 SFB 调用一些重要的系统功能，有存储区
系统功能（SFC）	集成在 CPU 模块中，通过 SFC 调用一些重要的系统功能，无存储区
功能块（FB）	用户编写的包含经常使用的功能的子程序，有存储区
功能（FC）	用户编写的包含经常使用的功能的子程序，无存储区
背景数据块（DI）	调用 FB 和 SFB 时用于传递参数的数据块，在编译过程中自动生成数据
共享数据块（DB）	存储用户数据的数据区域，供所有的块共享

组织块 OB 是操作系统和用户程序之间的界面，用于控制扫描循环和中断程序的执行、PLC 的启动和错误处理等，其他的组织块有启动组织块和中断组织块。档次越高的 CPU 可以使用的同类型组织块也就越多。操作系统只调用组织块，其他的程序块需要通过用户程序中的指令调用，操作系统才会加以处理（扫描）。其中最主要的组织块是 OB1，它是用户程序的主程序，也是操作系统自动做循环扫描的唯一的一个块。CPU 的操作系统完成启动过程后就循环执行 OB1，此时就可以在 OB1 中调用其他逻辑块。如果出现中断事件，当前正在执行的组织块只有在当前指令执行完以后才会被停止，操作系统才会分配给该事件一个组织块。这个中断事件（组织块）执行完后，被中断的块将会从断点处继续执行。

功能 FC 和功能块 FB 是用户程序中主要的逻辑操作块，二者皆是用户编写的子程序，只是功能（FC）不需要背景数据块，无存储区，调用结束数据消失；而功能块始终存在如影随形的背景数据块，有存储区，主要的控制、运算、操作等均由它们来完成。组织块负责安排 FC 和 FB 的调用条件和调用顺序。对于功能块，操作系统为参数及静态变量分配的存储空间是背景数据块。操作系统在 L 堆栈中给 FB 的临时变量分配空间；对于功能，操作系统同样也在 L 堆栈中给 FC 的临时变量分配空间，只是 FC 不能使用静态变量。输入、输出、I/O 参数以指向实参的指针形式存储在操作系统为参数传递而保留的额外空间中。

系统功能块 SFB 和系统功能 SFC 是集成在 S7 CPU 的操作系统中，预先编好程序的逻辑块，它们不占用程序空间。其中系统功能块通过 SFB 调用系统功能，有专用的存储区（即背景数据块），而系统功能通过 SFC 调用系统功能，没有专用的存储区。用户不能打开它们，也不能修改它们内部的程序。通常用于完成一些通用的功能，如读写实时时钟、设置参

数、数据通信等。在 S7-300CPU 中通常会固化有一部分 SFB 和 SFC，用户在编程时可以调用。通常 SFC 和 SFB 提供一些系统级的功能调用，比如 SFC39 “DIS_ IRT” 用于禁止中断和异步错误处理，可以禁止所有的中断、有选择地禁止某些范围的中断和某个中断；SFC40 “EN_ IRT” 用于激活新的中断和异步错误处理，可以允许所有的中断和有选择地允许某些中断等。在调用 SFB 时，需要用户指定背景功能块（CPU 中不包含其背景数据块），并确定将背景数据块下载到 PLC 中。

数据块 DB 只用于存储用户数据。在数据块中只有数据和变量，没有程序。但数据块占用程序容量。数据块中没有 STEP 7 的指令，STEP 7 按数据生成的顺序自动地为数据块中的变量分配地址。数据块可以分为全局（共享）数据块和背景（伴随）数据块两种。

应首先生成功能块，然后生成它的背景数据块。在生成背景数据块时指明它的类型为背景数据块（Instance）和它的功能块的编号。

（2）块的调用

在系统上电启动之后，首先开始运行初始化程序 OB100，之后进入可编程的工作周期，OB1 是循环扫描的主程序块，它的优先级最低。各种块的关系如图 7-1 所示。组织块 OB 可以调用 SFB、SFC、FB 和 FC；而 FB 和 FC 也可以调用另外的 FB 和 FC，称为嵌套。FB 和 SFB 使用时需要配有相应的背景数据块（IDB）。

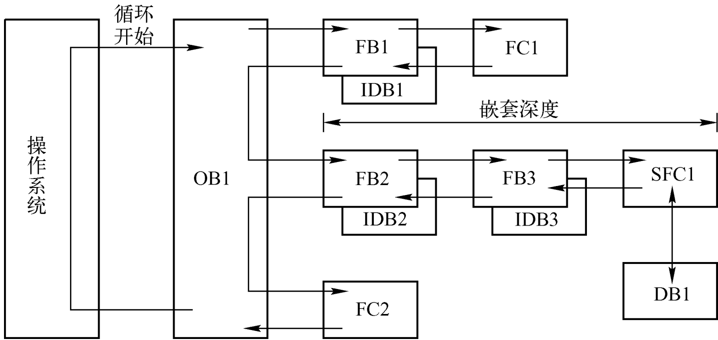


图 7-1　块的调用关系

（3）用户程序使用的堆栈

堆栈是 CPU 中的一块特殊的存储区，堆栈采用“先入后出”的规则存入和取出数据。最上面的存储单元称为栈顶。要保存的数据从栈顶“压入”堆栈时，堆栈中原有的数据依次向下移动一层，最下面的存储单元的数据丢失。在取出栈顶的数据后，堆栈中所有的数据依次向上移动一层。堆栈的这种“先入后出”的存取顺序，刚好满足块调用时（包括中断处理的块调用）存储和取出数据的要求，因此，堆栈在计算机的程序设计中的得到了广泛的应用。

① 局部数据堆栈（L 堆栈）

各逻辑块都有它的局部变量（L）存储区，局部变量在逻辑块的变量声明表中生成，只在它被创建的块中有效。局域数据堆栈用于存储块的局域数据区的临时变量、组织块的启动信息、块传递参数的信息和梯形图程序的中间结果。每个组织块用 20B 的临时局部数据来存储它的启动信息。局部数据可以按位、字节、字和双字来存取，例如 L0. 0、LB9、LW4

和 LD52。

CPU 分配给当前正在处理的块的临时局部数据的存储容量是有限的，这一存储区的大小和 CPU 的型号有关。CPU 给每一个优先级分配了局部数据区，这样可以保证不同优先级的 OB 都有它们可以使用的局部数据空间。S7-300CPU 每一优先级的局部数据区的大小都是固定的。

② 块堆栈（B 堆栈）

如果一个块的处理因为调用另外一个块，或者被更高优先级的 OB 块中止，CPU 将在块堆栈中存储以下消息。

a. 被中断的块的类型（OB、FB、FC、SFB、SFC）、编号和返回地址。

b. 从 DB 和 DI 寄存器中获得的块被中断时打开的共享数据块和背景数据块的编号。

c. 局部数据堆栈的指针。

利用这些数据，可以在中断它的任务处理完后回复被中断的块的处理。在多重调用时，堆栈可以保存参与嵌套调用的几个块的信息。图 7-2 给出了堆栈中数据的动态变化的情况。图中 OB1 调用功能 FC2，FC2 的执行被电源故障组织块 OB1 中断。

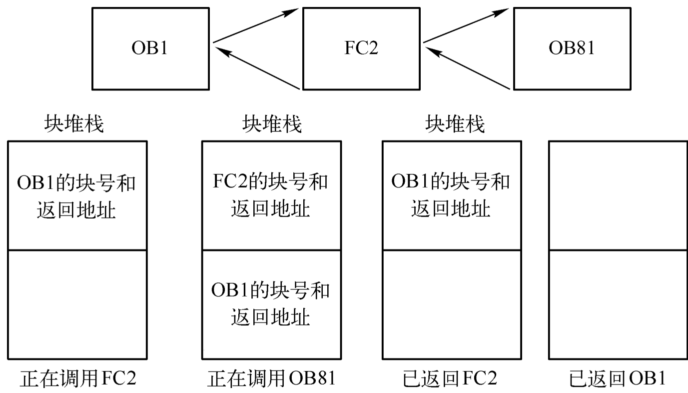


图 7-2　块堆栈

CPU 处于 STOP 模式时，可以在 CPU 的模块信息对话框中，查看堆栈保存的进入 STOP 模式时没有处理完的块，在块堆栈中，信息按照它们被处理的顺序存储。

③ 中断堆栈（I 堆栈）

如果程序的执行被优先级更高的 OB 中断，操作系统将保存下述寄存器的内容：当前的累加器和地址寄存器的内容、数据块寄存器 DB 和 DI 的内容、局部数据的指针、状态字、MCR（主控继电器）寄存器和块堆栈的指针。新的 OB 执行完后，操作系统读取中断堆栈中的信息，从被中断的地方开始继续执行程序。

CPU 在 STOP 或 HOLD 模式时，可以用 STEP 在 CPU 的模块信息中查看中断堆栈保存的数据，由此找出使 CPU 进入 STOP 模式的原因。

7.2　结构化编程的实现

结构化编程就是把一些过程要求类似或相关的功能进行分类、归纳、抽象化，然后编写一个与之相关的通用程序，这个程序就是一个结构。这种结构可以多处、多次使用。利用各种结构组合成程序的方法，就是结构化编程的方法。

7.2.1　结构化编程

结构化编程是对应于一些典型的控制要求编写通用的程序块，这些程序块可以反复被调用以控制不同的目标。这些通用的程序块就称为结构，利用各种结构来组成程序就成为结构化编程。要实现结构化编程有两个必要条件：一是程序能够分割，二是能够实现参数赋值。S7 程序是由块组成的，程序块也可以实现参数赋值，所以可以实现结构化。结构化编程除了可以避免上述缺点外，还有许多优点。它使程序通用化、标准化，简化了程序设计过程，缩短了程序的长度，减少了代码长度，减少编程工作量，提高了编程效率。

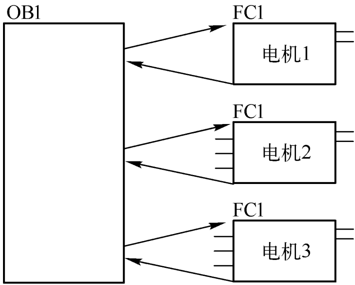


图 7-3　结构化编程

图 7-3 所示为编一个控制电机的通用程序（如 FC1），在主程序 OB1 中多次调用以控制不同的电机。

7.2.2　实现形式

以图 7-4 电机单向起动/停止的程序为例，其中 I0.0 为启动按钮，I0.1 为停止按钮，Q4.0 控制电机的接触器。该程序只能完成针对某个电机特定的控制。

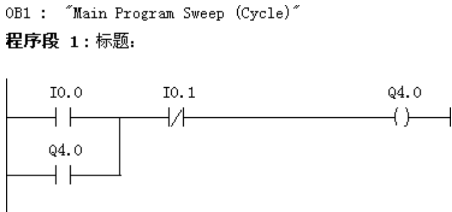


图 7-4　电机单向启动/停止程序

现在建一个功能 FC1，在 FC1 的变量声明表里声明：Start 为 IN 参数，数据类型为 Bool；Stop 为 IN 参数，数据类型为 Bool；Motor 为 OUT 参数，数据类型为 Bool。然后，用变量名（Start、Stop、Motor）代替原来的地址，如图 7-5 所示。

这样，FC1 就变成了电机单向起动/停止的通用程序，也就成了一个结构。这一结构在程序中可以被多次调用，在每次调用中再指定具体的控制目标，如图 7-6 所示。图中，FC1 框中的变量称为形式参数，在框外填上的地址称为实际参数。PLC 在运行中每次调用 FC1 时，把实际参数代到形式参数中进行运算，称为参数赋值。

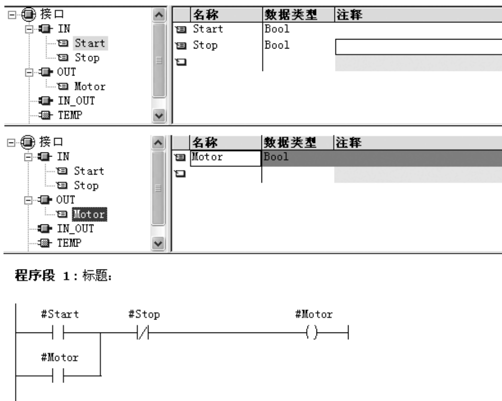


图 7-5　参数化的编程

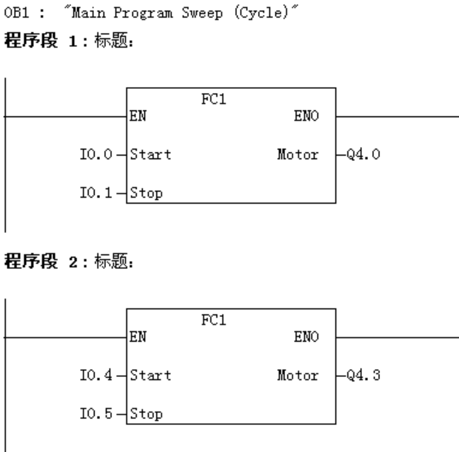


图 7-6　OB1 中多次调用 FC1 的程序

凡是通用的、典型的程序，都可以进行参数化、结构化。在 STEP 7 中工作的顺序是：在 FC、FB 的变量声明表中规范变量，也就是声明变量的名称、变量的类型和变量的数据类型。在编写程序的时候，不使用实际地址，而使用变量的名称（形式参数），得到一个普通的程序。在调用这些通用程序（结构）的时候，利用参数赋值的方法指定实际的控制条件和控制目标。

【例 7-1】设计故障信息显示电路，其故障信息显示的控制要求如图 7-7 所示。

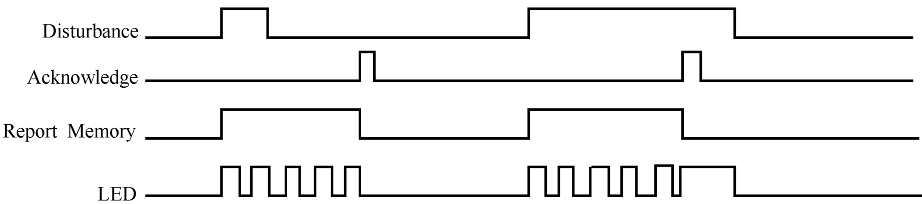


图 7-7　故障信息指示的时序

当故障信号“Disturbance”出现的时候，故障显示 LED 闪烁。值班人员看到后，按应答按钮“Acknowledge”，此时，如果故障已经排除，故障显示 LED 熄灭；如果故障尚未排除，则故障显示 LED 长亮。很短的故障信息也需要显示，为此，利用故障信号“Disturbance”的上升沿将“Report Memory”信号置 1，作为 LED 的工作标志。应答信号“Acknowledge”将这个工作标志复位（清 0）。如果按照故障信号“Disturbance”和显示信号 LED 的实际地址来编程，这个程序只能对付一个故障。把它参数化、结构化后，这就是故障显示的通用程序。建立故障处理的功能 FC10 并填写变量声明表，见表 7-2。

表 7-2　FC10 的变量声明表

地　址	参　数	名　称	类　型	初 始 值	说　明
0. 0	in	Disturb_input	BOOL		
0. 1	in	Acknowledge	BOOL		
0. 2	in	Flash_freq	BOOL		
2. 0	out	Display	BOOL		
4. 0	in_out	Edge_men_bit	BOOL		
4. 1	in_out	Report_memory	BOOL		

变量声明表中的参数，有 in、out、in_out 三种参数类型，其中：in 为输入参数，其值需要在被调用时从外部输入，在本块程序中是只读变量；out 为输出参数，其值需要送出外部，在本块程序中是只写变量；in_out 输入/输出参数，可以从外部输入也可以送出外部，是读/写变量。

变量声明表中还有临时变量 temp，用于存储一些中间结果。对于一个变量，除了要声明它的变量、参数类型之外，还要声明它的数据类型，可以用的数据类型与数据块的相同。

按照控制要求，利用变量声明表中提供的参数，编写程序，如图 7-8 所示。

程序中，变量值符号前面有“#”的，说明其是本块变量（Local Variable），而全局变量（Global Variable，在符号表中定义）的变量名上会有双引号。如果全局变量和本块变量没有重名，系统会自动辨别；如果有重名，系统首先会认为是本块变量，如果不是则需要用

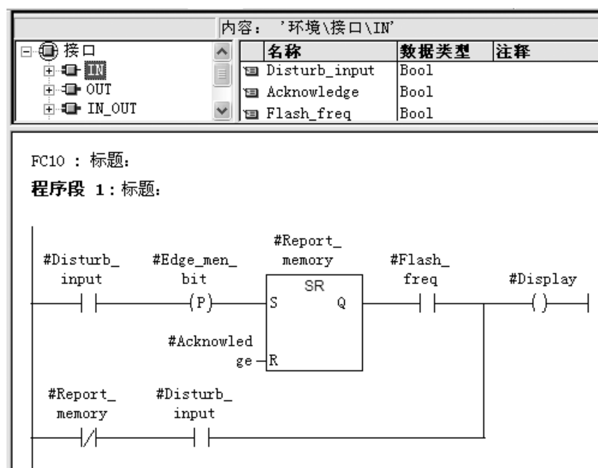


图 7-8　故障信息显示程序

户在输入的时候自己加上双引号。

现在，FC10 就可以被多次调用了。在 OB1 中两次调用 FC10 的例子如图 7-9 所示。

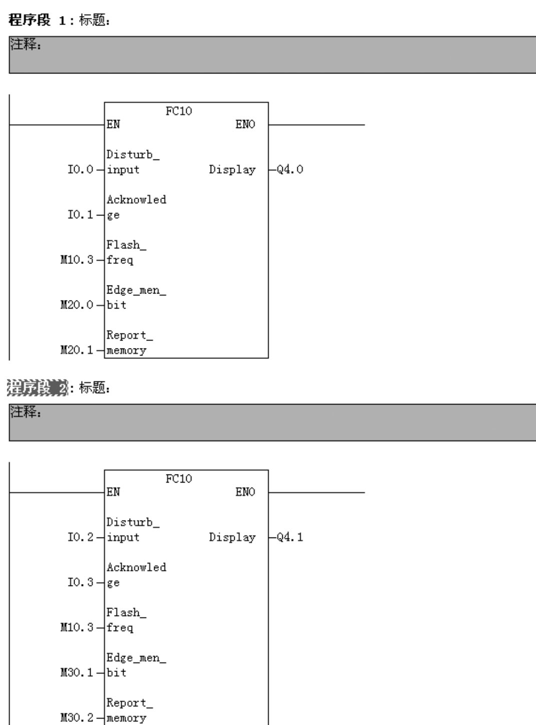


图 7-9　OB1 中调用 FC10

程序中的 M10.3 是一个闪烁的信号。可以自己利用定时器指令软件设计，也可以利用系统 CPU 属性的 Clock Memory，其具体使用参照硬件组态。

在变量声明表中把 Report_memory 和 Edge_men_bit 两个变量放在 in_out 变量中，是因为这两个变量在程序中要进行读、写。在调用 FC10 时，这两个变量就作为形式参数出现。

临时变量 temp 用来存储中间结果，变量值不需要从外面送出去。同时，变量 Report_memory 和 Edge_men_bit 不可以放在临时变量中。因为临时变量存储在 L 堆栈（Local Stack）中。当 FC10 执行完毕，返回主调块时，在 L 堆栈中的 FC10 的临时变量就会被清除。也就是说，FC 中的临时变量不能记住上一次调用时的数值；而变量 Report_memory 和 Edge_men_bit 在上一个扫描周期所取得的状态是必须记住的。程序中的 MW110 就是典型的中间变量，用临时变量来代替可以节省全局变量。

S7-300 PLC 的 L 堆栈的大小（表 7-3）。

表 7-3　S7-300 PLC 的 L 堆栈

执行的程序		对于 S7-300 PLC	
		优先级	局部堆栈大小
启动程序（只执行一次）	27		256
循环扫描程序	1		
时间中断	日时钟中断	2	256
	延时处理中断	3	256
	循环处理中断	12	256
事件驱动中断	硬件中断	16	256
	启动过程中的错误处理中断	28	256
	循环扫描中的错误处理中断	26	

L 堆栈的大小影响到程序调用时嵌套的深度（如图 7-10 所示）。

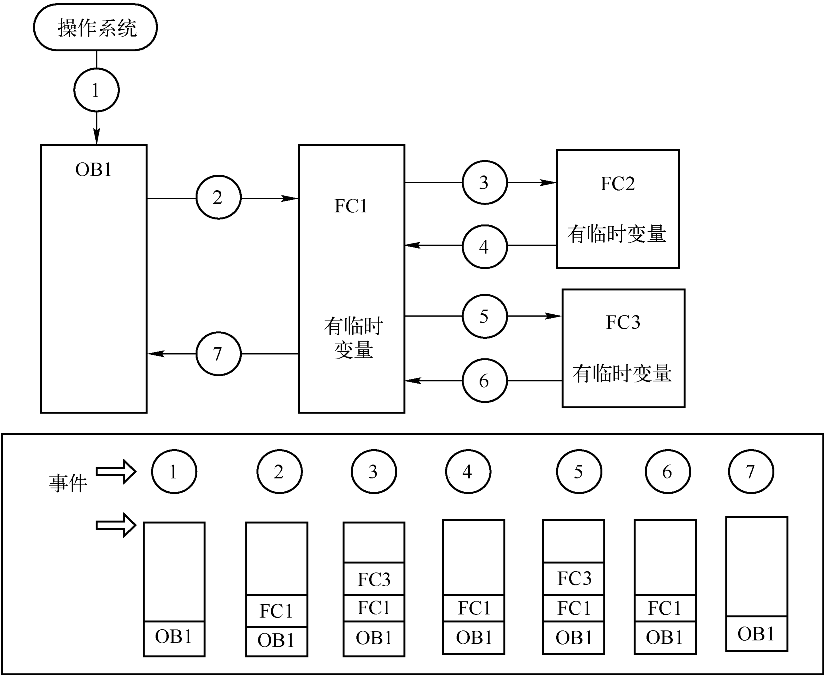


图 7-10　程序运行中的 L 堆栈的分配

在程序调用中，如果临时变量的数据超过了 L 堆栈的允许长度，CPU 进入 STOP 状态，并且，把停机原因“STOP caused by error when allocating data”写入诊断缓冲区 diagnostics buffer 中。

7.3 功能与功能块

功能（FC）、功能块（FB）和组织块（OB）统称为逻辑块（或程序块）。

功能块（FB）有一个数据结构与该功能块的参数完全相同的数据块，称为背景数据块，背景数据块依附于功能块，它随着功能块的调用而打开，随着功能块的结束而关闭。存储在背景数据块中的数据在功能块结束时继续保持。而功能（FC）则不需要背景数据块，功能调用结束后数据不能保持。

7.3.1 功能块

（1）生成功能块

如图 7-11 所示，选中 SIMATIC 管理器左边窗口中的“块”图标，用右键单击右侧窗口，执行出现快捷菜单中的“插入新对象”→“功能块”命令，生成一个新的功能块。在出现的功能块属性对话框中，采用系统自动生成的功能块的名称“FB1”，选择梯形图为默认的编程语言。如图 7-12 所示，单击“多重背景功能”前面的复选框，使小框中的“勾”消失。单击“确认”键后返回 SIMATIC 管理器，可以看到右边窗口中新生成的功能块 FB1。

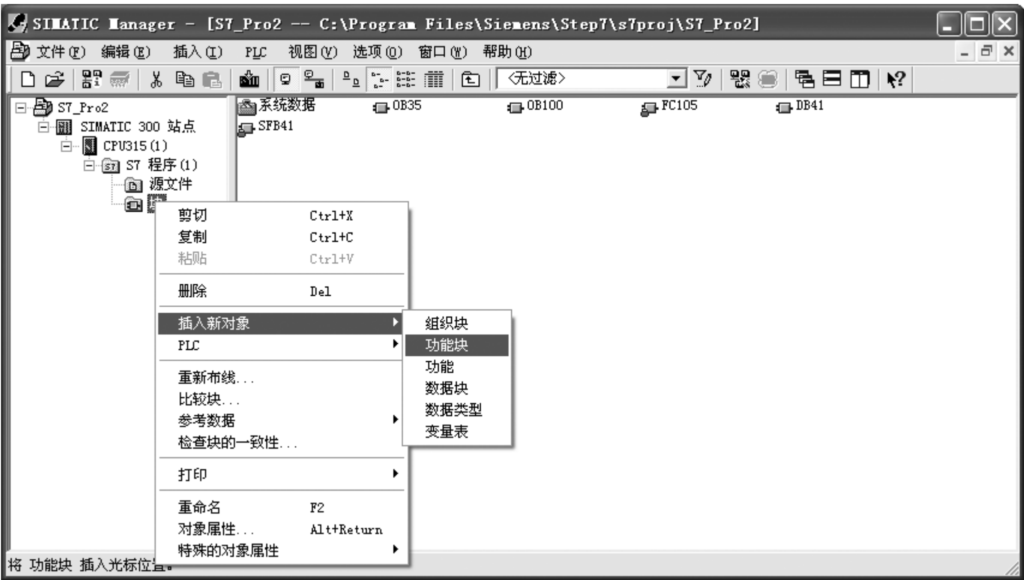


图 7-11 功能块生成

（2）局部变量

双击生成的“FB1”，打开程序编辑器。将鼠标的光标放在右侧程序区最上面的分隔条上，按住鼠标左键，往下拉动分隔条，分隔条上面是功能块的变量声明表，下面是程序区，左侧是指令列表框和库。将水平分隔条拉至程序编辑器视窗的顶端，不再显示变量声明表，



图 7-12　功能块常规参数设置

但是它依然存在。在变量声明表中声明块专用的局部变量，局部变量只能在它所在的块中使用。

变量声明表的左侧窗口给出了该表的总体结构，选中某一变量类型，例如“STAT”，在表的右侧显示的是输入参数“earlyValue”等的详细情况。具体设置如图 7-13 所示。

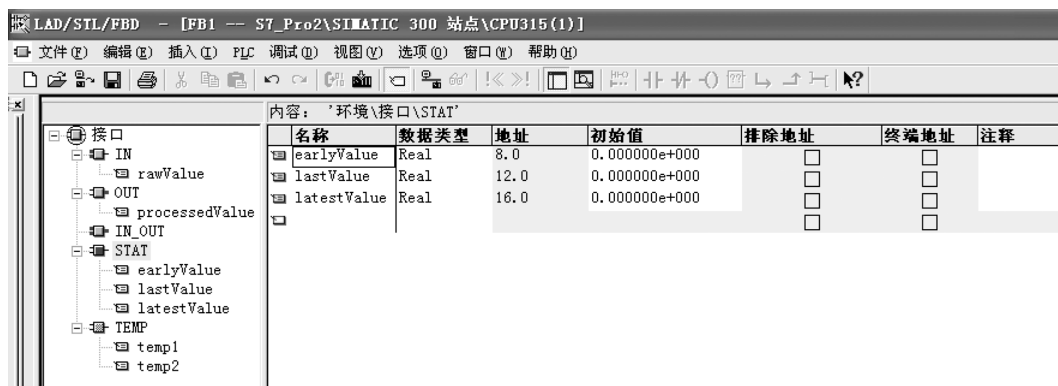


图 7-13　功能块形式参数设置

由图 7-13 可知，功能块有 5 种局部变量。

① IN：输入参数，用于将数据从调用块传送到被调用块。

② OUT：输出参数，用于将块的执行结果从被调用块返回给调用它的块。

③ IN_OUT（输入_输出参数）：参数的初值由调用它的块提供，块执行后由同一个参数将执行结果返回给调用它的块。

④ TEMP（临时变量）：暂时保存在局部数据区中的变量。只是在执行块时使用临时变量，执行完后，不再保存临时变量的数值，它可能被同一优先级中别的块里临时数据覆盖。临时变量区（L 堆栈）相当于无人管理的公告栏，任何人都可以往上面贴告示，后贴的告示将原来的告示覆盖掉。

⑤ STAT（静态变量）：从功能块执行完，到下一次重新调用它，静态变量的值保持不变。在功能块的背景数据块中使用。关闭功能块后，其静态数据保持不变。功能（FC）没

有静态变量。

选中变量声明表左侧窗口中的输入参数“IN”，在右侧窗口中生成两个 BOOL 变量和一个 INT 变量。用类似的方法生成其他局部变量，FB1 的背景数据块中的变量与变量声明表中的局部变量（不包括临时变量）相同。

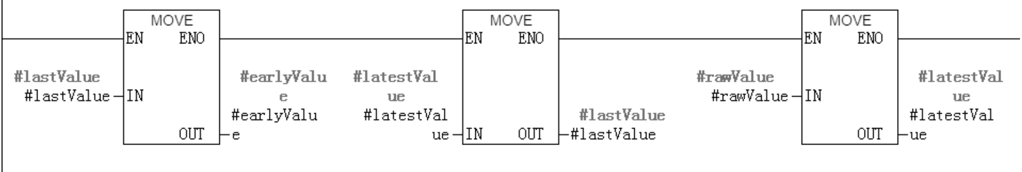
块的局部变量名必须以字母开始，只能由英文字母、数字和下划线组成，不能使用汉字，但是在符号表中定义的共享数据块的符号名可以使用其他字符（包括汉字）。

在变量声明表中赋值时，不需要指定存储器地址；根据各变量的数据类型，程序编辑器各自为所有的局部变量指定存储器地址。

块的输入参数、输出参数的数据类型可以使用基本数据类型、复杂数据类型、Timer（定时器）、Counter（计数器）、块（FB、FC、DB）、Pointer（指针）和 ANY 等。

（3）生成梯形图程序

图 7-14 的程序是功能块 FB1 的梯形图程序。其实现的功能是将最近采样的 3 个值先求和，再除以 3 作为当前采样值，一般用于工业上对采集的模拟量进行滤波处理。其中 rawValue 表示要处理的原始数据，earlyValue 表示 3 个数中最早采集的数据，lastValue 表示 3 个数中较早采集的数据，latestValue 表示 3 个数中最近采集的数据，processedValue 表示处理后的数据，temp1 与 temp2 是计算中中间结果。程序先将较早采集的数据放入最早采集数据的单元中，接着将最近采集的数据放入较早采集的数据单元中，再将要处理的原始数据放入最近采集的数据单元中，这样就把最新采集的要处理的原始数据与前 3 个数据中的后两个数据组合在一起构成了最新的 3 个数据，原来 3 个数据中最早的数据被覆盖，不再用。在此基础上对 3 个数据进行了相加，其中经过了两次加法，结果存在 temp1 与 temp2 中，最后，对 3 个数据之和除以 3，将处理后的结果放入了 processedValue 中。



程序段 2：标题：

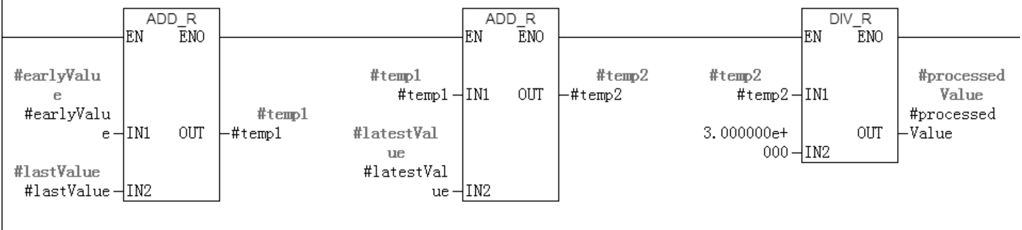


图 7-14 功能块 FB1 的梯形图程序

（4）背景数据块

由于功能块带有存储区，有一个称为背景数据块的局部数据块被分配给功能块，当调用功能块时，实际参数的值存在背景数据块中。如果在块调用时，没有实际参数分配给形式参数，则在程序执行中采用上一次存储在背景数据块中的参数值。

背景数据块用于保存 FB 和 SFB 的输入参数、输出参数、IN_OUT 参数和静态数据，背景数据块中的数据是自动生成的。它们是功能块变量声明表中的变量，临时变量存储在局部数据堆栈中。每次调用功能块时应指定不同的背景数据块。背景数据块相当于每次调用功能块时对应被控对象的私人数据仓库，它保存的数据不受其他逻辑块的影响。

功能块的数据保存在它的背景数据块中，功能块执行完后也不会丢失，以供下次执行时使用。其他逻辑块可以访问背景数据块中的变量。用户不能直接删除和修改背景数据块中的变量，只能在它对应的功能块的变量表中删除和修改这些变量。

使用不同的背景数据块调用功能块，可以控制多个同类的对象。生成功能块后，可以首先生成它的背景数据块，然后在调用该功能块时使用它。选中 SIMATIC 管理器左侧窗口中的“块”图标，用右键单击右边的窗口，执行出现的快捷菜单中的“插入新对象”→“数据块”命令，生成一个新的数据块。在出现的数据块属性对话框中，选择数据块的类型为“背景数据块”，如果有多个功能块，还需要设置它是哪一个功能块的背景数据块。功能块的变量声明表决定了它的背景数据块的结构和变量。

选中 SIMATIC 管理器左侧窗口中的“块”图标，用右键单击右边窗口，执行出现的快捷菜单中的“插入新对象”→“数据块”命令，生成一个新的数据块，其设置如图 7-15 所示。

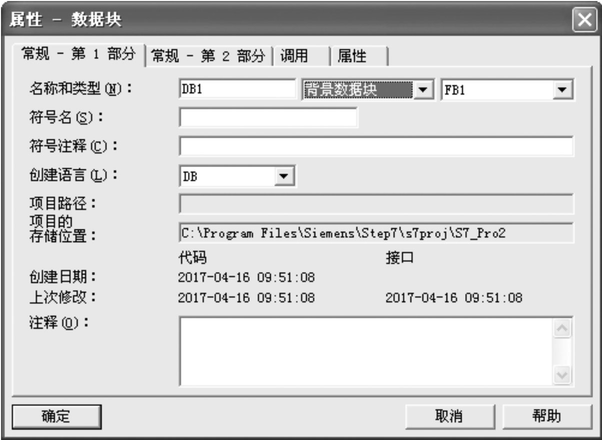


图 7-15　数据块参数设置

背景数据块中保存的数据就是在功能块中设置的形式参数，打开建立的数据块，其结果如图 7-16 所示。值得注意的是，背景数据块中的数据只能读，不能编辑。

DB 参数 - [DB1 -- S7_Pro2\SIMATIC 300 站点\CPU315(1)]

数据块(A)　编辑(E)　PLC(P)　调试(D)　查看(V)　窗口(W)　帮助(H)

	地址	声明	名称	类型	初始值	实际值	备注
1	0.0	in	rawValue	REAL	0.000000e+000	0.000000e+000	
2	4.0	out	processedValue	REAL	0.000000e+000	0.000000e+000	
3	8.0	stat	earlyValue	REAL	0.000000e+000	0.000000e+000	
4	12.0	stat	lastValue	REAL	0.000000e+000	0.000000e+000	
5	16.0	stat	latestValue	REAL	0.000000e+000	0.000000e+000	

图 7-16　背景数据块

生成功能块的输入参数、输出参数和静态变量时，它们被自动指定一个初始值，可以修改这些初始值。它们被传送给 FB 的背景数据块，作为同一个变量的初始值，调用 FB 时没

有指定实参的形参使用背景数据块中的初始值。

7.3.2　功能

如果逻辑块执行完后不需要保存它内部的数据，可以用功能（FC）来编程。与功能块（FB）相比较，FC 不需要配套的背景数据块。

（1）生成功能

用鼠标右键单击 SIMATIC 管理器左侧窗口中的“块”，执行出现的快捷菜单中的“插入新对象”→“功能”命令，生成一个新的功能。在出现新的功能属性对话框中，采用系统自动生成的功能的名称“FC1”，选择梯形图为功能默认的编程语言。功能生成及其参数设置分别如图 7–17 与图 7–18 所示。

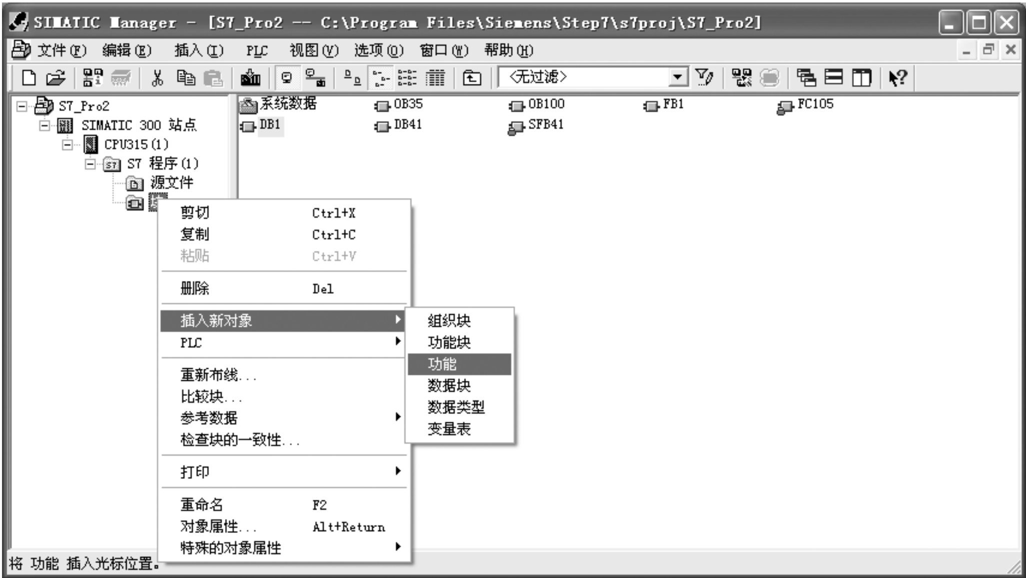


图 7–17　功能生成

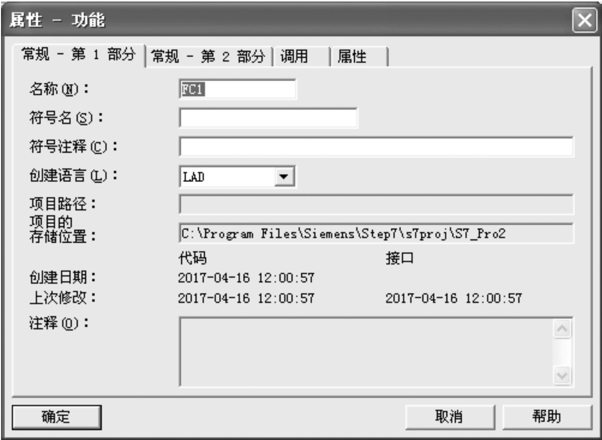


图 7–18　功能参数设置

（2）功能的局部变量

双击 SIMATIC 管理器中的“FC1”的图标，打开程序编辑器，设置功能 FC1 的形式参数如图 7-19 所示。

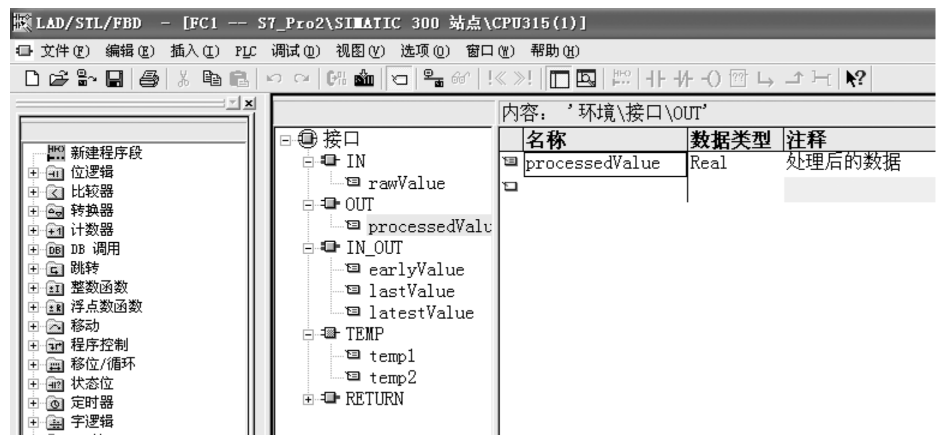


图 7-19　功能 FC1 的形式参数设置

仍以前述工业上对采集的模拟量进行滤波处理为例，将最近采样的 3 个模拟量值先求和，再除以 3 作为当前采样值，FC1 程序的编写如图 7-20 所示。

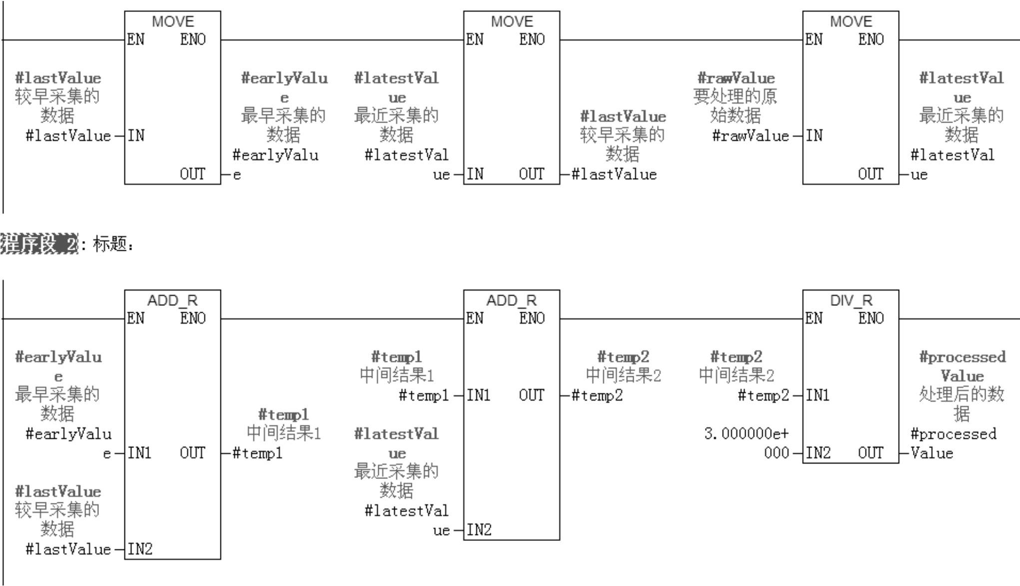


图 7-20　功能 FC1 程序

与功能块的变量声明表相比，功能没有静态变量（STAT），退出 FC 后不能保存它的临时局部变量。功能多了一个返回值 RET_VAL，它实际上是一个输出参数。返回值的设置与 IEC6113－3 标准有关，该标准的功能没有输出参数，只有一个返回值。在内部程序上，两者看不出有什么区别。

(3）功能与功能块的区别

FB 和 FC 均为用户编写的子程序，局部数据均有 IN、OUT、IN_OUT 和 TEMP，临时变量 TEMP 储存在局部数据堆栈中。

① FC 的返回值 RET_VAL 实际上是输出参数，因此，有无静态变量是两者的局部变量的本质区别，功能块的静态变量用背景数据块来保存。功能如果执行完后需要保存的数据，只能存储在全局变量（I/Q、PI/PQ、M、T、C 和共享数据块）中，但是这样会影响功能的可移植性。如果功能或功能块的内部不使用全局变量，只使用局部变量，不需要做任何修改，就可以将块移植到其他项目。如果块的内部使用了全局变量，在移植时需要重新统一分配它们内部使用的全局变量的地址，以保证不会出现地址冲突。当程序很复杂，子程序和中断程序很多时，这种重新分配全局变量地址的工作量很大，也很容易出错。如果逻辑块有执行完后需要保存的数据，显然应使用功能块，而不是功能。

② 功能块的输出参数不仅与来自外部的输入参数有关，还与用静态变量保存的内部状态数据有关。功能因为没有静态变量，相同的输入参数产生的执行结果是相同的。

③ 功能块有背景数据块，功能没有背景数据块。用户只能在功能内部访问功能的局部变量，其他逻辑块和人机界面可以访问背景数据块中的变量。

④ 不能给功能的局部变量设置初始值，可以给功能块的局部变量设置初始值。在调用功能块时，如果没有设置某些输入参数的实参，将使用背景数据块中的初始值，或上一次执行后的值。调用功能应给所有的形参指定实参。

7.3.3 功能的调用

功能和功能块的调用必须用实参代替形参，因为形参是在功能或功能块的变量声明表中定义的。为保证功能或功能块对同一类设备的通用性，在编程中不能使用实际对应的存储区地址参数，而是使用抽象参数，这就是形参。而块在调用时，必须将实际参数（实参）替代形参，从而可以通过功能或功能块实现对具体设备的控制。

块调用分为条件调用和无条件调用。用梯形图调用块时，块的 EN（Enable，使能）输入端有能流流入时执行块中的程序，反之则不执行。条件调用时使能输入端 EN 受到触点电路的控制。块被正确执行时 ENO（Enable Output，使能输出端）为 1，反之为 0。

双击打开 SIMATIC 管理器中的 OB1，在梯形图显示方式，将左侧窗口中的“FC 块”文件夹中的 FC1 拖入到程序段 1 的水平导线上，无条件调用符号名为“采集数据滤波”的 FC1。如图 7-21 所示。

方框的左侧是块的输入参数和输入/输出参数，右侧是输出参数。方框内的 rawValue、earlyValue 等是 FC1 的变量声明表中定义的形式参数，简称为形参。方框外的 MD10、MD14 等是形参对应的实际参数，简称为实参。调用功能或功能块时应将实参赋值给形参，并保证实参与形参的数据类型一致。

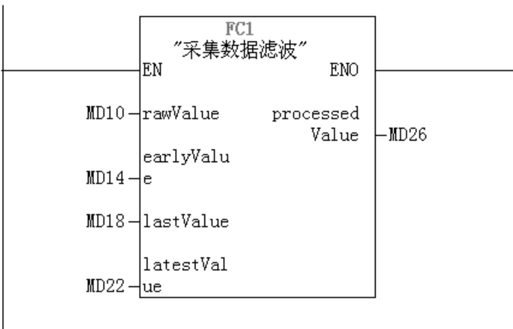


图 7-21　在 OB1 中调用 FC1

输入参数的实参可以是绝对地址、符号地址或常数，输出参数或输入_输出参数的实参必须指定为绝对地址或符号地址。功能 FC1 可以多次调用，将不同的实参赋值给形参，就可以实现对类似的但是不完全相同的被控对象的控制。

7.3.4 功能块的调用

将 OB1 左侧窗口中的“FB 块”文件夹中的 FB1 图标拖放到程序 1 的水平导线上，如图 7-22所示。FB1 的符号名为“发动机控制”。方框内的 rawValue、processedValue 是 FB1 的变量声明表中定义的输入、输出参数（形参）。方框外的 MD30、MD40 等是方框内的形参对应的实参。在调用块时，CPU 将实参分配给形参的值存储在背景数据块中。如果调用时没有给形参指定实参，功能块使用背景数据块中形参的数值。该数值可能是在功能块的变量声明表中设置的形参的初始值，也可能是上一次调用时存数在背景数据块中的数值。

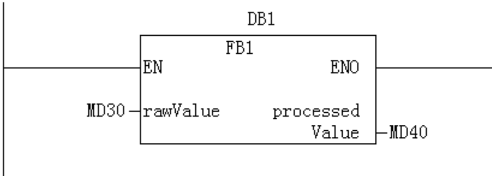


图 7-22 在 OB1 中调用 FB1

在 FB1 方框的上面，可以输入已经生成的 FB1 的背景数据块 DB1，也可以输入一个不存在的背景数据块，例如 DB2。输入后按〈Enter〉键，出现提示信息“实例数据块 DB2 不存在，是否要生成它?”，单击“是”按钮确认，可以在 SIMATIC 管理器中看到新生成的 DB2。

功能块 FB1 同样可以多次被调用，使用不同的实参和不同的背景数据块，使 FB1 分别用于任务相同的控制环节，区别仅在于变量的值（即实参的值）不同。

就此例而言，我们关心的只是 3 个采集数据的平均值，对 3 个采集数据本身并不关心。当用功能编程时，需要为 3 个采集数据提供全局地址进行保存，这样做给编程带来一定的麻烦，而用功能块编程时，只需为关心的数据提供全局地址即可，调用时显得简洁。

7.4 结构化程序设计实例

下面通过几个实例详细介绍结构化程序设计的思想和使用方法。

7.4.1 编辑并调用有参功能（FC）

有参功能（FC）是指编辑功能（FC）时，在局部变量声明表内定义了形式参数，在功能（FC）中使用了虚拟的符号地址完成控制程序的编程，以便在其他块中能重复调用有参功能（FC）。这种方式一般应用于结构化程序编写。

【例 7-2】多级分频器控制程序设计。

本例拟在功能 FC1 中编写二分频器控制程序，然后在 OB1 中通过调用 FC1 实现多级分频器的功能。多级分频器的时序关系如图 7-23 所示。其中 I0.0 为多级分频器的脉冲输入端；Q4.0～Q4.3 分别为 2、4、8、16 分频的脉冲输出端；Q4.4～Q4.7 分别为 2、4、8、16 分频指示灯驱动输出端。

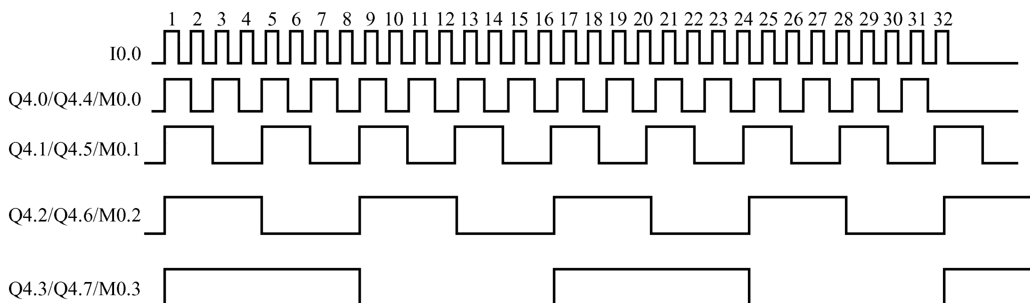


图 7-23　多级分频器控制时序图

（1）首先编辑有参功能（FC）

具体操作步骤如下。

① 创建多级分频器的 S7 项目：使用菜单“File”→“‘New Project’ Wizard”创建多级分频器的 S7 项目，并命名为“有参 FC”。

② 硬件配置：在“有参 FC”项目内打开“SIMATIC 300 站点”文件夹，双击硬件配置图标打开硬件配置窗口，并按图 7-24 所示完成硬件配置。



SIMATIC 300(1) (配置) -- 有参FC

(0) UR

2	CPU 315
3	
4	DI32xDC24V
5	DO32xDC24V/0.5A

(0) UR

插...	模块	订货号	固件	MPI 地址	I 地址	Q 地址	注释
1	PS 307 5A	6ES7 307-1EA00-0AA0					
2	CPU 315	6ES7 315-1AF03-0AB0		2			
3							
4	DI32xDC24V	6ES7 321-1BL00-0AA0			0...3		
5	DO32xDC24V/0.5A	6ES7 322-1BL00-0AA0				4...7	

图 7-24　硬件配置

③ 编写符号表：打开 S7 程序文件夹，双击符号编辑器图标将其打开，编辑符号表，如图 7-25 所示。

S7 程序(3) (符号) -- 有参FC\SIMATIC 300(1)\CPU 315

	状态	符号	地址		数据类型	注释
1		二分频器	FC	1	FC 1	对输入信号二分频
2		In_port	I	0.0	BOOL	脉冲信号输入端
3		F_P2	M	0.0	BOOL	2分频器上升沿检测标志
4		F_P4	M	0.1	BOOL	4分频器上升沿检测标志
5		F_P8	M	0.2	BOOL	8分频器上升沿检测标志
6		F_P16	M	0.3	BOOL	16分频器上升沿检测标志
7		Cycle_Execution	OB	1	OB 1	主循环组织块
8		Out_port2	Q	4.0	BOOL	2分频器脉冲信号输出端
9		Out_port4	Q	4.1	BOOL	4分频器脉冲信号输出端
10		Out_port8	Q	4.2	BOOL	8分频器脉冲信号输出端
11		Out_port16	Q	4.3	BOOL	16分频器脉冲信号输出端
12		LED2	Q	4.4	BOOL	2分频信号指示灯
13		LED4	Q	4.5	BOOL	4分频信号指示灯
14		LED8	Q	4.6	BOOL	8分频信号指示灯
15		LED16	Q	4.7	BOOL	16分频信号指示灯
16						

图 7-25　多级分频器符号表

④ 规划程序结构：按结构化方式设计控制程序，如图 7-26 所示，结构化的控制程序由两个逻辑块构成，其中 OB1 为主循环组织块，FC1 为二分频器控制程序。

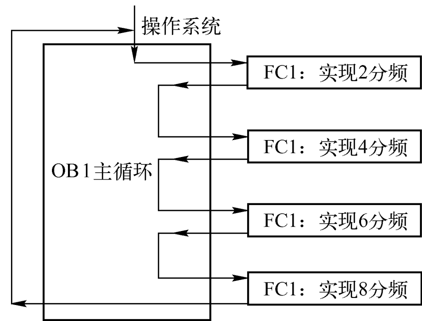


图 7-26　多级分频器程序结构

⑤ 创建有参 FC1：选择“有参 FC”项目的“Blocks”文件夹，然后执行菜单命令“Insert”→“S7 Block”→“Function”，在块文件夹内创建一个功能，并命名为“FC1”。由于在符号表内已经为 FC1 定义了符号，因此在创建 FC 的属性对话框内系统会自动添加符号表。

a. 编辑 FC1 的变量声明表：在 FC1 的变量声明表内，声明 4 个参数，表 7-4。

表 7-4　FC1 的变量声明表

接口类型	变量名	数据类型	注释
In	S_IN	BOOL	脉冲输入信号
Out	S_OUT	BOOL	脉冲输出信号
Out	LED	BOOL	输出状态指示
In_Out	F_P	BOOL	上跳沿检测标志

b. 编辑 FC1 的控制程序：二分频器的时序如图 7-27 所示。分析二分频器的时序图可以看到，输入信号每出现一个上升沿，输出便改变一次状态，据此可采用上跳沿检测指令实现。

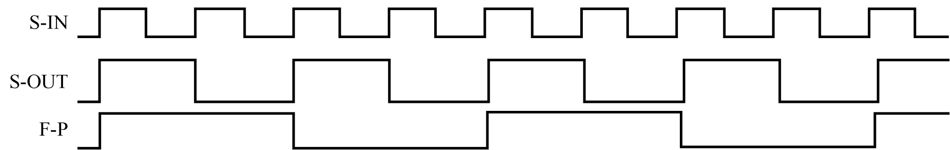


图 7-27　二分频器的时序图

如果输入信号 S_IN 出现上升沿，则对 S_OUT 取反，然后将 S_OUT 的信号状态送 LED 显示；否则，程序直接跳转到 LP1，将 S_OUT 的信号状态送 LED 显示。

在项目内选择“块”文件夹，双击 FC1，编写二分频的控制程序，如图 7-28 所示。

（2）其次在 OB1 中调用有参功能（FC）

在“有参 FC”项目内选择“块”文件夹，打开 OB1。在 LAD 语言环境下可以以块图的形式调用 FC1，如图 7-29 所示。

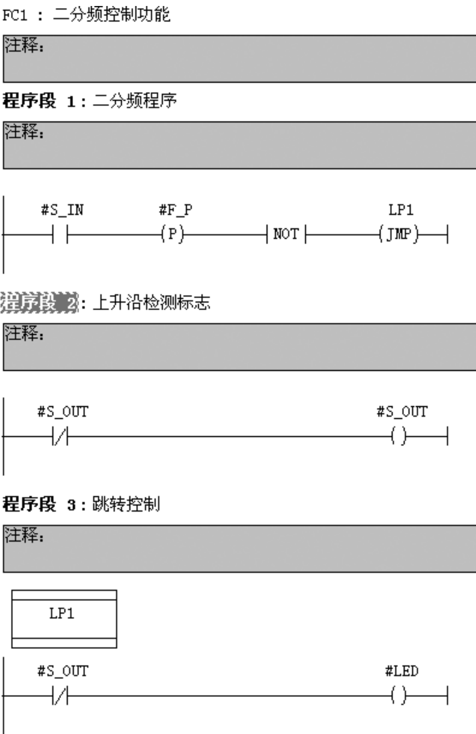


图 7-28　FC1 控制程序

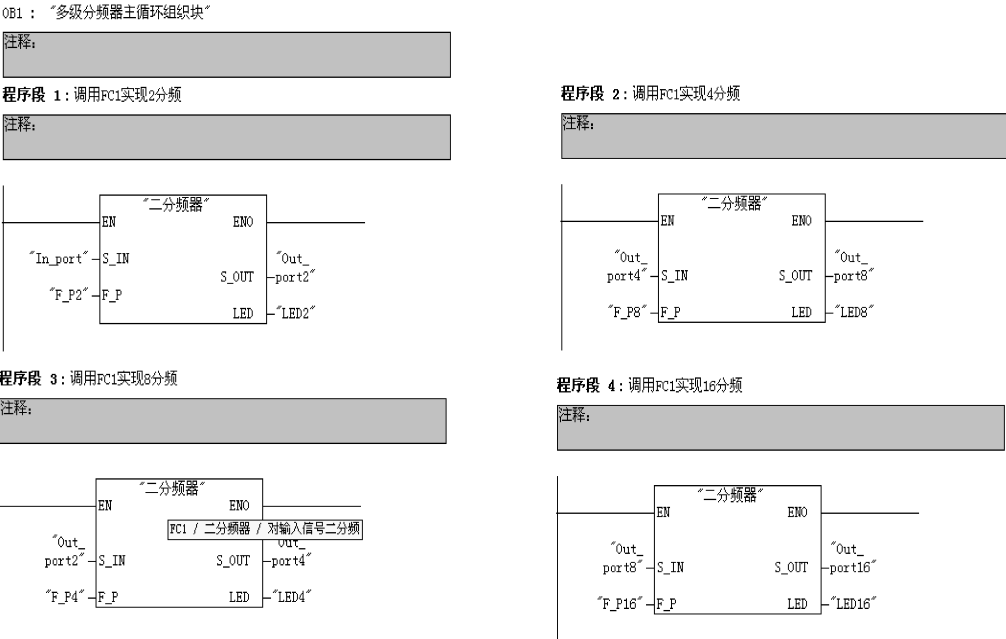


图 7-29　OB1 程序

7.4.2　多重背景使用举例

使用多重背景可以有效地减少数据块的数量，其编程思想是创建一个比 FB1 级别更高

的功能块，如 FB10，将未作任何修改的 FB1 作为一个“局部背景”，在 FB10 中调用。对于 FB1 的每一个调用，都将数据存储在 FB10 的背景数据块 DB10 中。

【例 7-3】 发动机组控制系统设计——使用多重背景。

（1）创建多重背景的 S7 项目

① 创建 S7 项目：使用菜单“File”→“‘New Project’ Wizard”创建发动机组控制系统的 S7 项目，并命名为“多重背景”。CPU 选择 CPU 315-2DP，项目包含组织块 OB1。

② 硬件配置：在“多重背景”项目内打开“SIMATIC 300 站点”文件夹，打开硬件配置窗口，并按图 7-30 所示完成硬件配置。



图 7-30 硬件配置

③ 编辑符号表：打开符号编辑器，编辑好的符号表如图 7-31 所示。

S7 程序(1) (符号) -- 多重背景\SIMATIC 300(1)\CPU 315-2DP

	状态	符号	地址		数据类型		注释
1		Automatic_Mode	Q	4.2	BOOL		运行模式
2		Automatic_On	I	0.5	BOOL		自动运行模式控制按钮
3		DE_Actual_Speed	MW	4	INT		柴油发动机的实际转速
4		DE_Failure	I	1.6	BOOL		柴油发动机故障
5		DE_Fan_On	Q	5.6	BOOL		启动柴油发动机风扇的命令
6		DE_Foolow_On	T	2	TIMER		柴油发动机风扇的继续运行时间
7		DE_On	Q	5.4	BOOL		柴油发动机的启动命令
8		DE_Preset_Speed	Q	5.5	BOOL		显示“已达到柴油发动机的预设转速”
9		Engine	FB	1	FB	1	发动机控制
10		Engine_Data	DB	10	DB	10	FB10的实例数据块
11		Engines	FB	10	FB	10	多重实例的上层功能块
12		Fan	FC	1	FC	1	风扇控制
13		Main_Program	OB	1	OB	1	此块包含用户程序
14		Manual_On	I	0.6	BOOL		手动运行模式控制按钮
15		PE_Actual_Speed	MW	2	INT		汽油发动机的实际转速
16		PE_Failure	I	1.2	BOOL		汽油发动机故障
17		PE_Fan_On	Q	5.2	BOOL		启动汽油发动机风扇的命令
18		PE_Foolow_On	T	1	TIMER		汽油发动机风扇的继续运行时间
19		PE_On	Q	5.0	BOOL		汽油发动机的启动命令
20		PE_Preset_Speed	Q	5.1	BOOL		显示“已达到汽油发动机的预设转速”
21		S_Data	DB	3	DB	3	共享数据块
22		Switch_Off_DE	I	1.5	BOOL		关闭柴油发动机
23		Switch_Off_PE	I	1.1	BOOL		关闭汽油发动机
24		Switch_On_DE	I	1.4	BOOL		启动柴油发动机
25		Switch_On_PE	I	1.0	BOOL		启动汽油发动机
26							

图 7-31 发动机组控制系统符号表

④ 规划程序结构：程序结构如图 7-32 所示，FB10 为上层功能块，它把 FB1 作为其“局部实例”，通过二次调用本地实例，分别实现对汽油机和柴油机的控制。这种调用不占用数据块 DB1 和 DB2，它将每次调用（对于每个调用实例）的数据存储到体系的上层功能块 FB10 的背景数据块 DB10 中。

图 7-32　发动机组控制系统程序结构

（2）编辑功能（FC）

FC1 用于实现发动机（汽油机或柴油机）的风扇控制，按照控制要求，当发动机起动时，风扇应立即起动；当发动机停机后，风扇应延时关闭。因此 FC1 需要一个发动机起动信号、一个风扇控制信号和一个延时定时器。

① 定义局部变量声明表：局部变量声明表见表 7-5，表中包含 3 个变量，其中两个 IN 型变量，1 个 OUT 型变量。

表 7-5　FC1 变量声明表

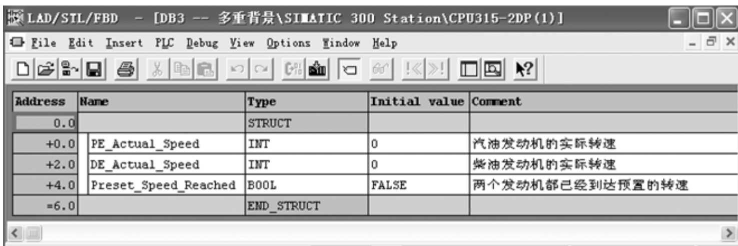
接 口 类 型	变　量　名	数 据 类 型	注　　释
In	Engine_On	BOOL	发动机起动信号
In	Timer_Off	Timer	用于关闭延迟的定时器功能
Out	Fan_On	BOOL	风扇起动信号

② 编辑 FC1 的控制程序：FC1 所实现的控制要求：发动机起动时风扇起动，当发动机再次关闭后，风扇继续运行 4 s，然后停止。定时器采用断电延时定时器，控制程序如图 7-33 所示。

图 7-33　FC1 的控制程序

（3）编辑共享数据块

共享数据块 DB3 可为 FB10 保存发动机（汽油机和柴油机）的实际转速，当发动机转速都达到预设速度时，还可以保存该状态的标志数据，如图 7-34 所示。



Address	Name	Type	Initial value	Comment
0.0		STRUCT		
+0.0	PE_Actual_Speed	INT	0	汽油发动机的实际转速
+2.0	DE_Actual_Speed	INT	0	柴油发动机的实际转速
+4.0	Preset_Speed_Reached	BOOL	FALSE	两个发动机都已经到达预置的转速
=6.0		END_STRUCT		

图 7-34　共享数据块

（4）编辑功能块（FB）

在该系统的程序结构内，有两个功能块：FB1 和 FB10。FB1 为底层功能块，所以应首先创建并编辑；FB10 为上层功能块，可以调用 FB1。

① 编辑底层功能块 FB1：在“多重背景”项目内创建 FB1，符号名“Engine”。功能块 FB1 的变量声明表见表 7-6。

表 7-6　FB1 的变量声明表

接口类型	变量名	数据类型	地址	初始值	注释
IN	Switch_On	BOOL	0. 0	FALSE	起动发动机
	Switch_Off	BOOL	0. 1	FALSE	关闭发动机
	Failure	BOOL	0. 2	FALSE	发动机故障,导致发动机关闭
	Actual_Speed	INT	2. 0	FALSE	发动机的实际转速
OUT	Engine_On	BOOL	4. 0	FALSE	发动机已起动
	Preset_Speed_Reached	BOOL	4. 1	FALSE	达到预置的转速
STAT	Preset_Speed	INT	6. 0	FALSE	要求的发动机转速

FB1 主要实现发动机的起停控制及速度监视功能，其控制程序如图 7-35 所示。

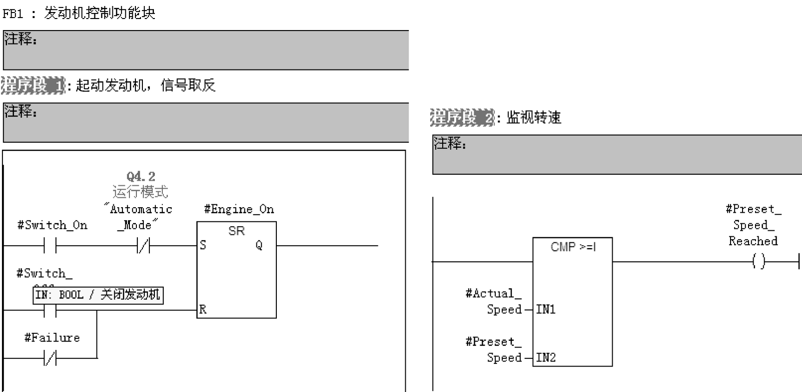


图 7-35　FB1 程序

② 编辑上层功能块 FB10：在“多重背景”项目内创建 FB10，符号名“Engines”。在 FB10 的属性对话框内激活“多重背景功能”选项，如图 7-36 所示。



图 7-36　将 FB10 设置成使用多重背景的功能块

要将 FB1 作为 FB10 的一个“局部背景”调用，需要在 FB10 的变量声明表中为 FB1 的调用声明不同名称的静态变量，数据类型为 FB1（或使用符号名“Engine”），如表 7-7 所示。

表 7-7　FB10 的变量声明表

接口类型	变量名	数据类型	地址	初始值	注释
OUT	Preset_Speed_Reached	BOOL	0.0	FALSE	两个发动机都已经达到预置的转速
STAT	Petrol_Engine	FB1	2.0	–	FB1“Engine”的第一个局部实例
	Diesel_Engine	FB1	10.0	_	FB1“Engine”的第二个局部实例
TEMP	PE_Preset_Speed_Reached	BOOL	0.0	FALSE	达到预置的转速（汽油发动机）
	DE_Preset_Speed_Reached	BOOL	0.1	FALSE	达到预置的转速（柴油发动机）

在变量声明表内完成 FB1 类型的局部实例“Petrol_Engine”和“Diesel_Engine”的声明以后，在程序元素目录的“Multiple Instances”目录中就会出现所声明的多重实例，如图 7-37 所示。接下来可在 FB10 的代码区，调用 FB1 的“局部实例”。

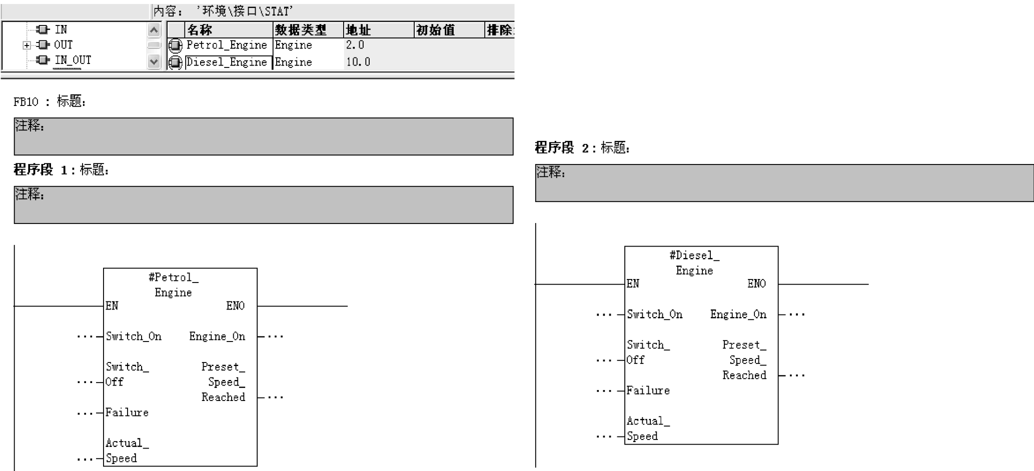


图 7-37　调用局部实例

编写功能块 FB10 的控制程序，如图 7-38 所示。调用 FB1 局部实例时，不再使用独立的背景数据块，FB1 的实例数据位于 FB10 的实例数据块 DB11 中。发动机的实际转速可直接从共享数据块中得到，如 DB3. DBW0（符号地址为“S_Data”. PE_Actual_Speed）。

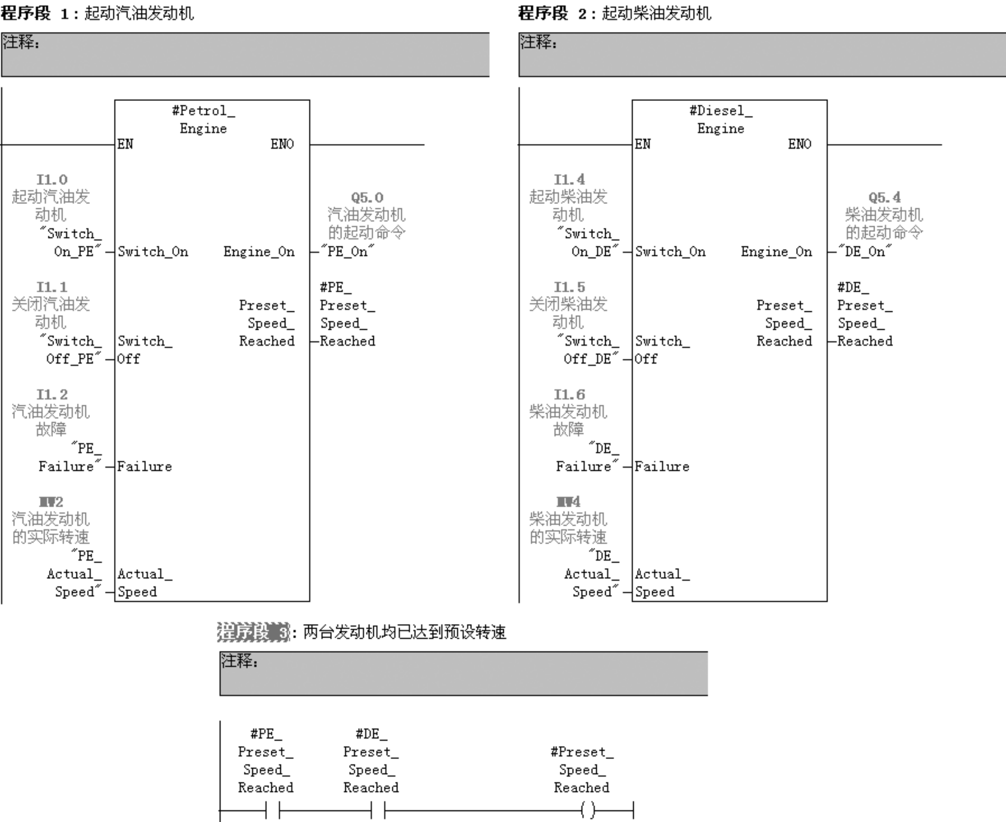


图 7-38　FB10 的控制顺序

（5）生成多重背景数据块 DB11

在“多重背景”项目内创建一个与 FB10 相关联的多重背景数据块 DB11，符号名“Engine_Data”，如图 7-39 所示。

DB 参数 - [DB11 -- 多重背景\SIMATIC 300(1)\CPU 315-2DP]

数据块(A)　编辑(E)　PLC(P)　调试(D)　查看(V)　窗口(W)　帮助(H)

	地址	声明	名称	类型	初始值	实际值	备注
1	0.0	out	Preset_Speed_Reached	BOOL	FALSE	FALSE	两个发动机都已经达到预置的转速
2	2.0	stat:in	Petrol_Engine.Switch_On	BOOL	FALSE	FALSE	启动发动机
3	2.1	stat:in	Petrol_Engine.Switch_Off	BOOL	FALSE	FALSE	关闭发动机
4	2.2	stat:in	Petrol_Engine.Failure	BOOL	FALSE	FALSE	发动机故障，导致发动机关闭
5	4.0	stat:in	Petrol_Engine.Actual_Speed	INT	0	0	发动机的实际转速
6	6.0	stat:out	Petrol_Engine.Engine_On	BOOL	FALSE	FALSE	发动机已开启
7	6.1	stat:out	Petrol_Engine.Preset_Speed_Reached	BOOL	FALSE	FALSE	达到预置的转速
8	8.0	stat	Petrol_Engine.Preset_Speed	INT	1500	1500	要求的发动机转速
9	10.0	stat:in	Diesel_Engine.Switch_On	BOOL	FALSE	FALSE	启动发动机
10	10.1	stat:in	Diesel_Engine.Switch_Off	BOOL	FALSE	FALSE	关闭发动机
11	10.2	stat:in	Diesel_Engine.Failure	BOOL	FALSE	FALSE	发动机故障，导致发动机关闭
12	12.0	stat:in	Diesel_Engine.Actual_Speed	INT	0	0	发动机的实际转速
13	14.0	stat:out	Diesel_Engine.Engine_On	BOOL	FALSE	FALSE	发动机已开启
14	14.1	stat:out	Diesel_Engine.Preset_Speed_Reached	BOOL	FALSE	FALSE	达到预置的转速
15	16.0	stat	Diesel_Engine.Preset_Speed	INT	1500	1500	要求的发动机转速

图 7-39　DB11 的数据结构

（6）在 OB1 中调用功能（FC）及上层功能块（FB）

OB1 控制程序如图 7-40 所示，在程序段 4 中调用了 FB10。

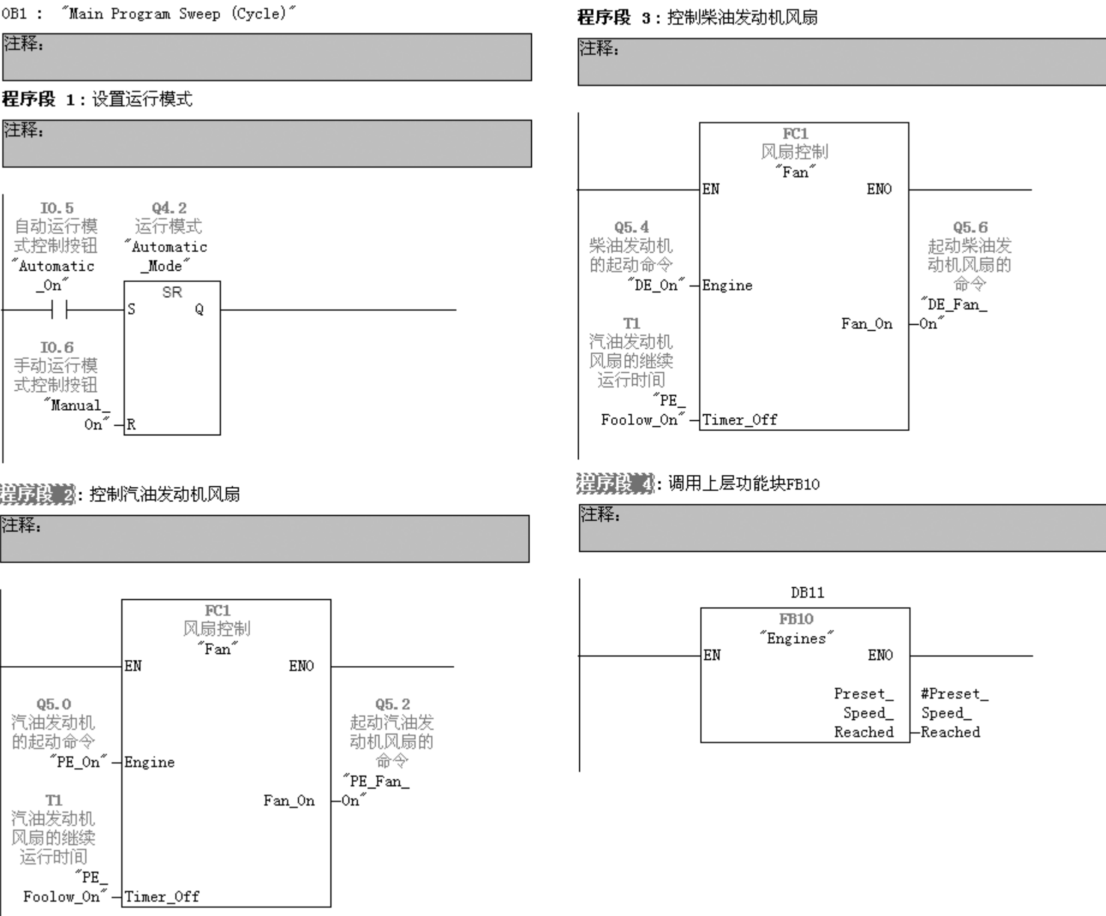


图 7-40　OB1 控制程序

习题

7-1　简述 FC 与 FB 的区别。

7-2　简述结构化编程与模块化编程以及线性化编程的区别。

7-3　简述多重背景的使用方法。

7-4　用结构化编程方法实现求 4 个数的算术平均值。

第 8 章　数据块和组织块

在 S7-300 CPU 中，用户程序是由启动程序、主程序和各种中断响应程序等不同的程序模块构成的，这些模块在 CPU 中的实现形式就是数据块（DB）组织块（OB）。数据块（DB）是用来存放执行用户程序时所需的数据与存储程序执行结果的数据存储区，其作用与标志寄存器类似，但数量更多。组织块（OB）是操作系统与用户程序的接口，块由操作系统调用，用于控制扫描循环和中断程序的执行、PLC 的启动和错误处理等，有的 CPU 只能使用部分组织块。

8.1　数据块

在西门子的可编程控制器中，数据是以变量的形式来存储的。有一些数据，如 I、Q、M、T、C 等，存在系统存数区内，而大量的数据存储在数据块。数据块（DB）用子分类存储设备或者生产线中变量的值，也是实现各逻辑块之间数据交换、数据传递和共享数据的重要途径。数据块丰富的数据结构便于提高程序的执行效率和进行数据库管理。

数据块占用程序容量，与逻辑块不同，数据块里只有变量声明部分的数据，而没有用户程序。从用户的角度出发，数据块主要有两个作用：其一是用来存储一些在设备运行之前就必须放到 PLC 中的重要数据，在运行过程中，用户程序主要是去读这些数据；其二是在数据块中根据需要安排好存储数据的位置和顺序，以便在生产过程中把一些重要的数据（如产量、实际测量值等）存储在这些指定的位置上。

不同的 CPU 允许建立的数据块的块数和每个数据块最多库占用的字节数是不同的，具体的参数可以查看 PLC 选型手册。

8.1.1　数据块的类型

STEP 7 按照数据块的使用方法把数据分为三类。

共享数据块（Shared DB）：共享数据块又称全局数据块，它不附属于任何逻辑块，用于存储全局数据。在共享数据块和全局符号表中声明的变量都是全局变量，所有逻辑块（OB、FB、FC）都可以访问共享数据块存储的信息。CPU 可以同时打开一个共享数据块和一个背景数据块。如果某个逻辑块被调用，它可以使用临时局域数据（即 L 堆栈）。逻辑块执行结束后，其局域数据区中的数据丢失，但是共享数据块中的数据不会被删除。

背景数据块（Instance DB）：背景数据块也称为“私有存储器区”，即用作功能块（FB）的存储器。FB 的参数和静态变量安排在其背景数据块中。背景数据块不是由用户编辑的，而是由编辑器生成的。背景数据块只能被指定的功能块访问。应首先生成功能块，然后生成其背景数据块。在生成背景数据块时，应说明它的类型为背景数据块，并指明其数据块的编号，如 FB2。背景数据块的功能块被执行完后，背景数据块中存储的数据不会丢失。在调用功能块时使用不同的背景数据块，可以控制多个同类的对象。

用户定义的数据块（DB of Type）：用户定义数据块是以 UDT 为模版所生成的数据块。创建用户定义数据块之前，必须先创建一个用户定义数据类型，如 UDT1，并在 STL/FBD/LAD S7 程序编辑器内定义。

背景数据块与共享数据块的区别在于：在背景数据块中不可以增加或删除变量，也不能改变默认和实际参数。如果在 FB 中增加或删除默认的参数或变量，必须更新 FB 所有的背景数据块或者删除并重新建立。在全局数据块中可以增加或删除变量，也可以改变默认或实际参数。

在 STEP 7 中数据块的数据类型可以采用基本数据类型、复杂数据类型或用户定义数据类型（UDT）。

① 基本数据类型：根据 IEC1131 - 3 定义，长度不超过 32 位（即不超过累加器 ACCU 的长度），可利用 STEP 7 基本指令处理，能完全装入 S7 处理器的累加器中。基本数据类型包括位（BOOL）、字节（BYTE）、字（WORD）、双字（DWORD）、整数（INT）、双整数（DINT）和浮点数（FLOAT，或称实数 REAL）等。

② 复杂数据类型，就是基本数据类型的组合，只能结合共享数据块的变量声明使用。数据长度超过 32 位。因为数据长度超过累加器的长度，所以不可以一次性用装入指令把整个数据装入累加器中，一般利用库中的标准块（“IEC” S7 程序）处理复杂数据。复杂数据类型包括日期和时间（DATE_AND_TIME）、字符串（STRING）、数组（ARRAY）和结构（STRUCT）。

8.1.2 数据块的生成

在 STEP 7 中，为了避免出现系统错误，在使用数据块之前，必须先建立数据块，并在块中定义变量（包括变量符号名、数据类型及初始值等）。数据块中变量的顺序及类型决定了数据块的数据结构，变量的数量决定了数据块的大小。数据块建立后，还必须同程序块一起下载到 CPU 中，才能被程序访问。

建立数据块的方法和建立程序块的方法一样。在 SIMATIC Manager 窗口下，用鼠标右键单击 Blocks，然后选中“Insert New Object - Data Block”，就会弹出“Properties - Data Block”对话框，在对话框中的 Name and Type 栏中做出正确选择后，单击 OK 按钮，就建立了一个新的数据块。和打开程序块进行编辑一样，双击数据块图标，就把这数据块打开了，如图 8-1 所示。

DB1 -- S7_Pro2\SIMATIC 300 站点\CPU312C(1)

地址	名称	类型	初始值	注释
0.0		STRUCT		
+0.0	DB_VAR	INT	0	临时占位符变量
=2.0		END_STRUCT		

图 8-1　编辑数据块

刚打开的数据块是空的，用户必须自己编辑这个数据块。在名称栏目中填上变量名称，在类型栏目中填上数据类型。在类型栏目中可以用鼠标右键列出数据类型清单，然后选择合适的数据类型。名称和类型是必须填写的。系统会根据数据类型自动地为每个变量分配地址。这是一个相对地址，它相当重要，因为在程序中往往需要根据地址来访问这个变量。在初始值栏目，可以按需要填上初始值，也可以不填。若不填写，则初始值就为零；若填写了初始值，则在首次存盘时系统会将该值赋值到实际值栏中。下载数据块时，下载的值是实际值，初始值不能下载。注释栏目，填写该变量的注释，也可以让它空着。每个数据块的长度

取决于实际编辑的长度。而最大的长度，对于 S7-300 PLC 来说是 8 KB。

数据块有两种显示方式，即声明表显示方式和数据显示方式，分别用菜单命令“View > Declaration view”和“View > Data view”来指定这两种方式。

数据块编辑好后，一定要存盘、下载，不同的 CPU 允许建立的数据块的块数和每个数据块最多可以占用的字节数是不同的，具体的数量取决于 CPU 型号，可以查看选型手册。

8.1.3 数据块的访问

只有打开的数据块才能访问。由于 CPU 有两个数据块寄存器：DB 寄存器和 DI 寄存器，最多可以同时打开两个数据块。一个作为共享数据块，共享数据块的块号存储在 DB 寄存器中；另一个作为背景数据块，背景数据块的块号存储在 DI 寄存器中。没有关闭数据块的指令，打开下一个数据块时，就自动关闭原来的数据块。

在用户程序中可能存在多个数据块，而每个数据块的数据结构并不完全相同，因此在访问数据块时，必须指明数据块的编号、数据类型与位置。如果访问不存在的数据单元或数据块，而且没有编写错误处理 OB 块，CPU 将进入 STOP 模式。

按地址来访问数据块时，可以以 Bit 为单位、Byte 为单位、Word 为单位或者 DWord 为单位。同时，所访问的地址必须是已经编辑的，访问不存在的地址会导致出错。

在程序中访问的数据块，无非是读或者写数据块中的变量。要访问数据块首先要打开数据块，打开数据块的指令是：OPN DBXX。在 STEP 7 中可以采用传统访问方式，即先打开后访问；也可以采用完全表示的直接访问方式。数据块的访问方法如图 8-2 所示。

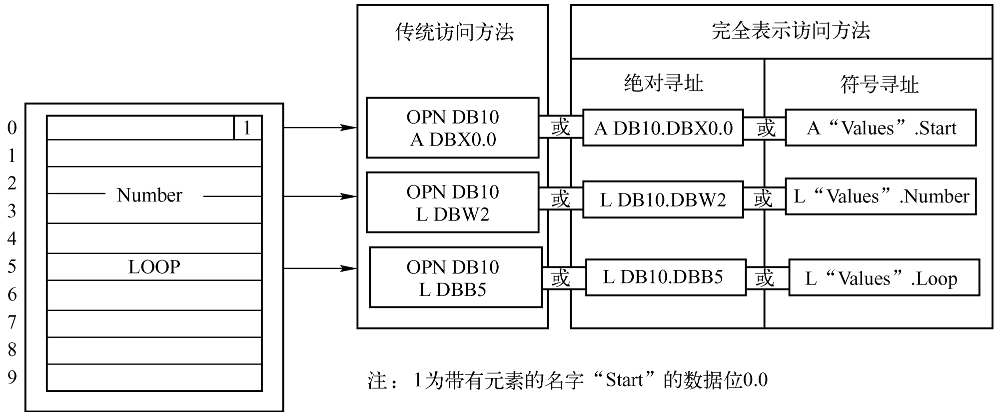


图 8-2　访问数据块

图 8-2 中 L DB10. DBW2 相当于 OPN DB10 和 L DBW2 两条指令。完全表示访问方法对于地址的指定完整而明确，最好采取这种方法。指定地址的时候，可以用绝对地址，也可以用符号地址。但要注意，当采用绝对地址时，数据块和变量都要用绝对地址，如 L DB10. DBW2；当采用符号地址时，数据块和变量都要用符号地址，如 L“Values”. Number，不可以两种地址混合使用。因此，要使用符号地址，就必须在符号表中给数据块指定符号。用符号访问数据块，程序的可读性比较高，而且修改数据也比较容易。

在功能（FC）调用时，系统会把当前数据块压入块堆栈，在返回主调块时，再把当前数据块弹出。当使用传统访问方法时，要特别注意所访问的变量是属于哪一个数据块的。块

调用时，数据块打开和关闭的关系如图 8-3 所示。

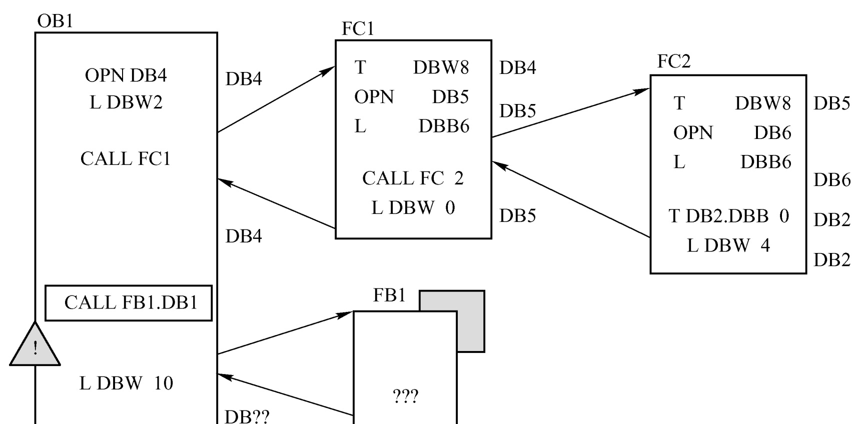


图 8-3　数据块的打开和关闭

图 8-3 中，从 FC2 返回 FC1 时，系统把调用指令（CALL FC2）时的当前数据块（DB5）弹出，所以指令 L DBW0 所指的数据块是 DB5；从 FC1 返回 OB1 时也类似。

功能块 FB 的调用与功能 FC 的调用不一样。因为 FB 有伴随数据块，调用功能块 FB 时伴随数据块会自动打开，放在 DI Register。返回主调块时不会打开原来的当前数据块；要访问数据块时，必须重新打开数据块。采用完全表示访问法可以避开上述的麻烦。

8.2　S7-300 PLC 的组织块

8.2.1　组织块的分类

组织块（OB）是操作系统与用户程序之间的接口，组织块由操作系统调用。组织块中的程序是由用户编写的。SIMATIC S7 CPU 提供了大量的 OB（组织块），组织块采用中断的方式在特定的时间或特定的情况下执行相应的程序和响应特定事件的程序。所以理解中断的工作过程及其相关的概念对组织块的编程有着重要的意义。

S7-300 PLC 在正常的情况下，PLC 按照循环扫描的工作方式执行用户程序。如果要对某些特殊的外部事件或内部事件进行快速响应，PLC 则采用中断的方式进行处理。

组织块只能由操作系统启动，它由变量声明表和用户编写的控制程序组成。当操作系统调用时，每个 OB 提供 20 个字节的变量声明表，其含义取决于 OB。变量名称是标准 STEP 7 规定的。组织块可以分为以下几类。

① 循环执行的组织块：需要连续执行的程序存储在 OB1 中，执行完后又开始新的循环。

② 启动组织块：启动组织块用于系统的初始化，CPU 上电或操作模式改为 RUN 时，S7-300 PLC执行 OB100，S7-400 PLC 根据组态的启动方式来执行 OB100～OB102 中的一个。

③ 定期执行的组织块：定期执行的组织块包括日期时间中断组织块（OB10～OB17）和循环中断组织块（OB30～OB38）。可以根据设定的日期时间或时间间隔来执行中断程序。

④ 事件驱动的组织块：事件驱动的组织块包括延时中断、硬件中断、异步错误中断和

同步错误中断。延时中断（OB20～OB23）在过程事件出现后延时一定的时间再执行中断程序；硬件中断（OB40～OB47）用于需要快速响应的过程事件，事件出现时马上中止当前正在执行的程序执行对应的中断程序；异步错误中断（OB80～OB87）和同步错误中断（OB121 和 OB122）用于决定在出现错误时系统如何响应。

⑤ 背景组织块：STEP 7 可以监视最大扫描循环时间，也可以保证设置最小扫描循环时间。如果在 OB1 中所有的嵌套中断和系统功能的执行时间，小于所设置的最小扫描循环时间，CPU 在执行完 OB1 后，尚有一段空闲时间，可执行背景组织块 OB90。用户可以将对运行时间要求不高的操作，放在 OB90 中执行，避免出现等待时间。背景组织块的优先级最低，其运行时间不受 CPU 操作系统的监视，在 OB90 中编写的程序长度不受限制。

组织块是操作系统调用的，OB 没有背景数据，也不能为 OB 声明输入、输出变量和静态变量，因此 OB 的变量声明表中只有临时变量。OB 的临时变量可以是基本数据类型、复合数据类型或数据类型 ANY。

操作系统为所有的组织块声明了一个 20B 的包含 OB 启动信息的变量声明表，见表 8-1。声明表中变量的具体内容与组织块的类型有关。用户可以通过 OB 的变量声明表获得与启动 OB 的原因有关的信息。

表 8-1　OB 的变量声明表

地址（字节）	内　　容
0	事件级别与标识符，例如 OB40 为 B#16#11，表示硬件中断被激活
1	用代码表示与启动 OB 事件有关的信息
2	优先级，例如 OB60 的优先级为 25
3	OB 块号，例如 OB60 的块号为 60
4～11	附加信息，例如 OB40 的第 5 个字节为产生中断模块的类型，16#54 为输入模块，16#55 为输出模块。第 6、7 字节组成的字为产生中断模块的起始地址，第 8～11 号字节组成的双字为产生中断的通道号。
12～19	启动 OB 的日期和时间（年、月、日、时、分、秒、毫秒和星期）

8.2.2　中断

S7 CPU 提供的各种不同的组织块采用中断的方式在特定的时间或特定的情况下执行相应的程序和响应特定事件的程序。理解中断的工作过程及相关概念对组织块的编程有着重要的意义。

（1）中断过程

启动事件触发 OB 调用称为中断，中断处理用于实现对特殊内部事件或外部事件的快速响应。在 SIMATIC S7 中，对这些特殊事件的处理，安排了大量的组织块，可在这些组织块中编写相应的中断处理程序。没有中断时，CPU 循环执行组织块 OB1，因为除背景组织块 OB90 以外，OB1 的中断优先级最低。当 CPU 检测到中断源的中断请求时，操作系统在执行完当前程序的当前指令（即断点处）以后，就会根据中断优先级的高低立即响应中断。CPU 暂停正在执行的程序，调用中断源对应的用于中断的组织块来处理。执行完中断组织块以后，则返回中断程序的断点处继续执行原来的程序。

当有中断发生时，如果没有下载对应的组织块，CPU 将会进入 STOP 模式。如果用户希望忽略某个中断事件，则可以生成和下载一个对应的空的组织块，出现该中断事件时，CPU 就不会进入 STOP 模式了。如果在执行中断程序时，又检测到一个中断请求，CPU 将比较两个中断源的中断优先级。如果优先级也相同，则按照产生中断请求的先后顺序来处理。如果后者的优先级高于正在执行的中断的优先级，将会中止当前正在执行的 OB，改为调用较高优先级的 OB（中断的优先级即是组织块的优先级）。这种处理的方式被称为中断程序的嵌套调用。

一个组织块的执行被另一个组织块中断时，操作系统会对现场进行保护。被中断的 OB 的局部数据压入 L 堆栈（局部数据堆栈），被中断的断点处的现场信息保存在 I 堆栈（中断堆栈）和 B 堆栈（块堆栈）中。

当中断发生时，中断程序是由操作系统自动调用的，而不是由程序块调用的。编写中断程序时，首先要遵循“越短越好”的原则，尽量减少中断程序的执行时间，以减少对其他处理的延迟，否则可能引起主程序控制的设备操作异常。其次是因为不能预知系统何时调用中断程序，中断程序不能改写其他程序中可能正在使用的存储器，所以不要轻易使用其他程序中可能使用的编程元件，而应尽量使用相应组织块的临时局域变量。

（2）中断的优先级

中断的优先级即是组织块的优先级，较高优先级的组织块可以中断较低优先级的组织块的处理过程。如果同时产生的中断请求不止一个，则最先执行优先级最高的 OB，然后按照优先级由高到低的顺序依次执行其他的组织块。

组织块的优先级由低到高的顺序是：背景循环、日期时间中断、时间延时中断、循环中断、硬件中断、多处理器中断、I/O 冗余错误、异步错误、启动和 CPU 冗余、同步错误和主程序扫描循环，其中 S7-300 PLC 的组织块的优先级是固定不变的。通常情况下组织块的号码越大，其优先级也就越高。表 8-2 所示为各种组织块的默认优先级。

表 8-2　组织块的默认优先级

中断类型	组　织　块	默认优先级
主程序扫描循环	OB1	1
日期时间中断	OB10 ~ OB17	2
延时中断	OB20 ~ OB23	3 ~ 6
循环中断	OB30 ~ OB38	7 ~ 15
硬件中断	OB40 ~ OB47	16 ~ 23
多处理器中断	OB60	25
同步循环中断	OB61 ~ OB64	25
I/O 和 CPU 冗余错误	OB70 I/O 冗余错误中断；OB72 CPU 冗余错误中断，例如，一个 CPU 发生故障；OB 73 通信冗余错误中断，例如，冗余连接的冗余丢失	25，28，25
异步错误	OB80 ~ OB87	26（在启动程序中则为 28）
背景循环	OB90	29（优先级最低）
启动	OB100 ~ OB102	27
同步错误	OB121 ~ OB122	与导致此错误的 OB 优先级相同

同一个优先级可以分配给好几个 OB，具有相同优先级的 OB 则按启动它们的事件出现的先后顺序来处理。被同步错误启动的故障 OB 的优先级与错误出现时正在执行的 OB 的优先级相同。在生成逻辑块 OB、FB 和 FC 时，同时生成临时局部变量数据，CPU 的局部数据区按优先级划分。

每个组织块的局部数据区都有 20B 的启动信息，它们只是在该块被执行时使用的临时变量，这些信息在 OB 启动时由操作系统提供，包括启动事件、启动日期与时间、错误及诊断事件。将优先级赋值为 0，或分配小于 20B 的局部数据给某一个优先级，可以取消相应的中断 OB。

8.2.3 启动组织块

用于启动时的组织块包括 OB100、OB101、OB102。S7 CPU 在处理用户程序前，要先执行一个启动程序，这就是操作系统要调用的启动组织块。

当 PLC 接通电源以后，CPU 有 3 种启动方式：热启动（Hot Restart）、暖启动（Warm Restart）和冷启动（Cold Restart）。不同的 CPU 具有不同的启动方式，可以在 STEP 7 中设置 CPU 的属性时选择其一。例如 S7-300 系列 PLC 中，除了 CPU318 可以选择暖启动或冷启动外，其他的 CPU 只有暖启动的方式；对于 S7-400 系列 PLC，可以根据不同的 CPU 型号，选择热启动、暖启动或者冷启动中的一种启动方式。

（1）暖启动

S7-300 CPU（不包括 CPU 318）只有暖起动。手动暖启动时，将 CPU 的模式选择开关扳到 STOP 位置，“STOP” LED 指示灯亮，然后再扳到 RUN 或 RUN－P 位置。暖启动时，过程映像数据及非保持的存储器位、定时器和计数器都被复位。具有保持功能的存储器位、定时器、计数器和所有的数据块将保留原数值。程序将重新开始运行，执行 OB100 后，循环执行 OB1。

（2）热启动（仅 S7-400 PLC 有）

如果 PLC 在运行期间突然断电，然后又重新上电，CPU 将执行一个初始化程序 OB101，自动完成热启动。热启动从上次 RUN 模式结束时程序被中断之处继续执行，不对计数器等复位。热启动只能在 STOP 状态时没有修改用户程序的条件下才能进行。

（3）冷启动

手动冷启动时将 CPU 的模式选择开关扳到 STOP 位置，“STOP” LED 指示灯亮，再扳到 MRES 位置，STOP 指示灯灭 1s，亮 1s，再灭 1s，然后常亮，最后将模式开关再扳到 RUN 或者 RUN－P 位置。自动冷启动时过程数据区的所有过程映像数据、存储器位、定时器、计数器、数据块以及有保持功能的器件的数据，都被复位为零。用户程序从装载存储器载入工作存储器，调用 OB102 后，循环执行 OB1。

发生下列事件时，CPU 执行启动功能：PLC 电源上电后；CPU 的模式选择开关从 STOP 位置拨到 RUN 或 RUN－P 位置时；接收到通过通信功能发送来的启动请求；多 CPU 方式同步之后和 H 系统连接好后（只适用于备用 CPU）。

启动用户程序之前，先执行启动组织块 OB。在热启动、暖启动和冷启动时，操作系统分别调用 OB100、OB101、OB102 组织块。用户可以通过在启动组织块 OB100～OB102 中编写程序来设置 CPU 的初始化操作。例如设置开始运行时某些变量的初始值以及输出模块的

初始值等。

启动程序没有长度和时间的限制，因为循环时间监视器还没有被激活。在启动程序期间不能执行时间中断程序和硬件中断程序。在设置 CPU 模块属性的对话框中，可以在“启动”选项卡中设置启动的各种参数。OB100 的变量声明表中各参数的含义见表 8-3。

表 8-3　OB100 声明表中变量的含义

变　　量	类　　型	描　　述
OB100_EV_CLASS	Byte	事件类型及标识符
OB100_STARTUP	Byte	启动方式
OB100_PRIORITY	Byte	OB 优先级
OB100_OB_NUMBR	Byte	OB 号
OB100_RESERVED_1	Byte	系统保留
OB100_RESERVED_2	Byte	系统保留
OB100_STOP	Word	导致 CPU 停机的事件
OB100_STRT_INFO	DWord	系统启动信息
OB100_DATE_TIME	Date_And_Time	OB100 启动的日期和时间

8.2.4　时间延时中断组织块

（1） 概述

在延时中断组织块中，用户可以编写将要延时的程序。PLC 中定时器的定时时间与扫描方式有关，其精度受不断变化的循环周期的影响，而使用延时中断组织块则可以提高延时的精度。延时中断组织块的延时时间为 1～60000 ms，其延时精度为 1 ms，优于定时器的精度。

S7 提供了 4 个延时中断组织块（OB20～OB23），CPU 可以使用的延时中断 OB 的个数与 CPU 的型号有关。S7-300 系列中 CPU318 能使用 OB20 和 OB21，其余 S7-300CPU 只能使用 OB20，以 OB20 为例来说明其用法。延时中断用 SFC 32“SRT_DINT”启动，延时时间是 SFC32 的一个输入参数，延时时间在 SFC 32 中设置，启动后经过设定的时间触发中断，调用 SFC 32 指定的 OB。如果延时中断移动被启动，延时时间还没有到达，延时中断可以用 SFC 33“CAN_DINT”取消。用 SFC 34“QRY_DINT”查询延时中断的状态。

当用户程序调用 SFC32（SRT_DINT）时，需要提供 OB 编号、延时时间和用户专用的标识符。经过指定的延时后，相应的 OB 将会启动。可使用 SFC39～SFC42 来禁用或延迟并重新使延迟中断。只有当 CPU 处于 RUN 模式下时才会执行延时 OB。暖重启或冷重启将清除延时 OB 的所有启动事件。

延时时间（单位为 ms）和 OB 编号一起传送给 SFC32，时间到期后，操作系统将启动相应的 OB。设置延时中断，最基本的步骤是：调用 SFC32（SRT_DINT），并将延时中断 OB 作为用户程序的一部分下载到 CPU。

如果以下任一情况发生，操作系统将调用一个异步错误组织块。

① 在调用 SFC32 时，OB20 并没有下载到 CPU 中。

② 一个延时中端的执行还没有结束，下一个延时中断又被启动。

OB20 的局部变量见表 8-4，这些变量的定义为用户编程提供了方便。其中变量“OB20

_PRIORITY”是代表 OB20 的优先级，默认为 3，可以通过设置这个变量参数改变优先级。

表 8-4　OB20 的局部变量

变　　量	类　　型	地　　址	描　　述
OB20_EV_CLASS	BYTE	0.0	事件级别和识别码，B#16#11：中断激活
OB20_STRT_INF	BYTE	1.0	B#16#20 ~ B#16#21；OB20 ~ OB23 的启动请求
OB20_PRIORITY	BYTE	2.0	分配的优先级：默认值为 3（OB20 ~ OB23）
OB20_OB_NUMBER	BYTE	3.0	OB 号（OB20 ~ OB30）
OB20_RESERVED_1	BYTE	4.0	保留
OB20_RESERVED_2	BYTE	5.0	保留
OB20_SIGN	WORD	6.0	用户 ID：SFC32 的输入参数 SING
OB20_DTIME	TIME	8.0	以毫秒形式组态的延时时间
OB20_DATE_TIME	DATE_AND_TIE	12.0	OB 被调用的日期和时间

（2）应用方法

首先可以在 STEP 7 中查看可以支持的延时中断 OB。具体方法如下：在 STEP 7 的硬件组态窗口中，双击项目机架上的 CPU 所在的行，打开 CPU 属性对话框，单击“中断”选项页，设置框中显示出当前 CPU 支持的延时中断组织块，如图 8-4 所示。

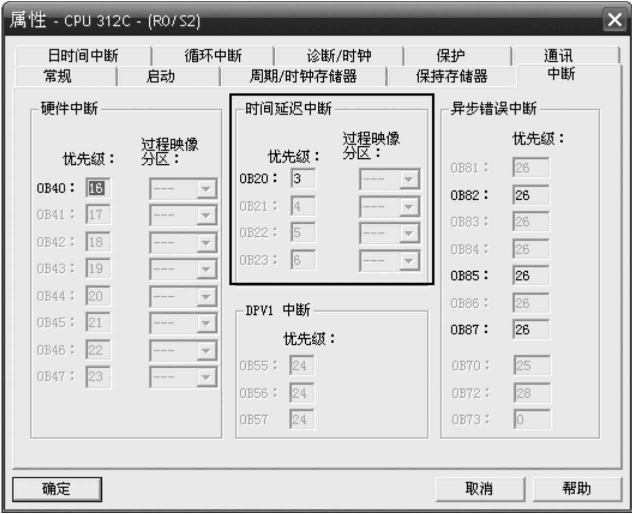


图 8-4　CPU 支持的延时中断组织块

延时中断组织块 OB20 经过一段指定时间的延时时间后运行。在程序中，系统提供了 SFC32 ~ SFC34 三个系统功能块供用户使用。OB20 在调用 SFC32“SRT_DINT”后启动，延时时间在 SFC32 的参数设定；如果在延迟时间未到之前想取消延时程序的执行，可以调用 SFC33“CAN_DINT”；同时可以使用 SFC34“QRY_DINT”查询延迟的状态。在应用中，必须将 OB20 和使用的 SFC 都下载到 PLC 中。表 8-5 所示为 SFC32 ~ SFC34 的说明。

表 8-5　SFC32 ~ SFC34 的参数说明

参　　数	声　　明	数 据 类 型	存　储　器	参 数 说 明
OB_NR	INPUT	INT	I、Q、M、D、L、常数	OB 号（OB20 ~ OB23），延时后被启动
DTIME	INPUT	TIME	I、Q、M、D、L、常数	延时值（1 ~ 60000 ms）
SING	INTPUT	WORT	I、Q、M、D、L、常数	当延时中断 OB 被调用时，在起始事件信息中出现的开始标志
RET_VAL	OUTPUT	INT	I、Q、M、D、L、	如故障发生，返回值包含故障代码
STATUS	OUTPUT	WORT	I、Q、M、D、L、	时间中断的状态

（3）应用实例

【例 8-4】 M1.0 控制启动延时中断 OB20，延时 10 s 后中断一次，中断程序使 MW20 加 1，M1.2 控制取消延时中断 OB20。其梯形图如图 8-5 的（1）~（4）所示。

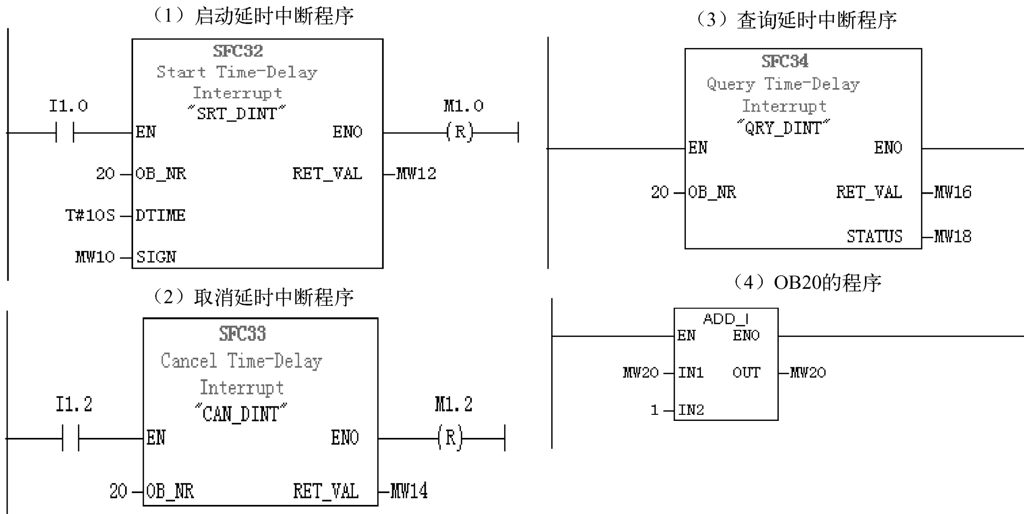


图 8-5　程序梯形图

说明：在图 8-5OB 的（1）中调用系统功能 SFC 启动延时中断，DTIME 端是延时的时间设置此时为 T#10 s。程序编译保存好后就可以下载到实际 PLC 或 PLCSIM 仿真软件中了。注意的是，要把所有的程序块都下载，包括 OB20、SFC32 等，选中要下载的程序块，再单击工具栏的下载按钮即可。

8.2.5　硬件中断组织块

（1）概述

硬件中断组织块（OB40 ~ OB47）用于快速响应信号模块 SM、通信处理模块 CP 和功能模块 FM 的信号变化。例如：当一个数字量输入模块的一个通道有上升沿信号来时，若这个模块有中断能力，则触发一个硬件中断，中断服务程序就置于硬件中断组织块中。

CPU318 只能使用 OB40 和 OB41，其余的 S7-300 CPU 只能使用 OB40（见表 8-6）。S7-400 CPU 可以使用的硬件中断 OB 的数量与 CPU 的型号有关。

表 8-6　OB40 的局部变量

变　　量	类　　型	描　　述
OB40_EV_CLASS	BYTE	事件级别和诊断号：B#16#11：中断被激活
OB40_STRT_INF	BYTE	B#16#41：中断通过中断行 1 B#16#42～B#16#44：中断通过中断行 2～4（只对 S7-400 PLC） B#16#45：WinAC 通过 PC 触发的中断
OB40_PRIORITY	BYTE	分配优先级：默认 16（OB40）～23（OB47）
OB40_OB_NUMBR	BYTE	OB 号（OB40～OB47）
OB40_RESERVED_1	BYTE	保留
OB40_IO_FLAG	BYTE	输入模块：B#16#54 输入模块：B#16#55
OB40_MDL_ADDR	WORD	触发中断模块的逻辑地址
OB40_POINT_ADDR	DWORD	数字模板：带有模板输入状态的位字段（0 位对应第一个输入） 模拟模板：带有限幅信息输入通道的位字段 CP 或 IM：模块中断状态（不是与用户有关的）
OB40_DATE_TIME	DATE_AND_TIME	被调用的日期和时间

用户可以使用 STEP 7 的硬件组态功能来决定信号模块哪一个通道在什么条件下产生硬件中断，将执行哪个硬件中断 OB，OB40 被默认用于执行所有的硬件中断。对于 CP 和 FM，可以在对话框中设置相应的参数来启动 OB。

只有用户程序中有相应的组织模块，才能执行硬件中断。否则操作系统将会向影响诊断缓冲区输入错误信息，并执行异步错误处理组织块 OB80。

硬件中断 OB 默认的优先级为 16～23，可以通过参数设置改变 S7-400 PLC 的优先级。

硬件中断被模块触发后，操作系统将自动识别是哪一个槽的模块和模块中哪一个通道产生的硬件中断。硬件中断 OB 执行完后，将发送通道确认信号。

如果在处理硬件中断的同时，又出现了其他硬件中断事件，新的中断按以下方法去识别和处理：如果正在处理某一中断事件，又出现了同一模块同一通道产生的完全相同的中断事件，新的中断事件将丢失，即不处理它。

（2）应用方法

首先可以在 STEP 7 中查看支持的硬件中断组织块。具体方法是：在 STEP 7 的硬件组态窗口中，双击项目中机架上的 CPU 所在的行，打开 CPU 属性对话框，单击“中断”选项页，可以看到 CPU 支持的硬件中断模块，如图 8-6 所示。在此也可以为硬件中断 OB 选择优先级。

通过 STEP 7 进行参数赋值，可以为能够触发硬件中断的每一个信号模块指定参数

（3）应用实例

【例 8-5】 I1.0 的上升沿作为硬件中断触发脉冲，使用硬件中断 OB40，当来一次 I1.0 的上升沿，就使 MW0 自动加 1。

首先在硬件组态中设置中断触发信号。并不是所有的信号模块都具有中断功能。此例中，需要一个数字量输入模块，图 8-7 所示为硬件组态，其右视图硬件目录中的“DI－300”中有此版本软件支持的所有 SM321，单击一个模块后，右下角处将出现这个模块的基本信息。然后插入“CPU313－2DP”和一块具有中断功能的数字量输入模块。然后双击模板，选择

图 8-6　CPU 支持的硬件中断块

“中断”选项，可同时激活“硬件中断”和“硬件中断触发器”选项，图 8-8所示设置数字量输入模块的中断。

图 8-7　硬件组态

说明：在这个例子中，也可以使用 SFC39 和 SFC40 来取消和激活中断。这里只设置中断模块，并在 OB40 中编程即可完成功能。图 8-9 所示为硬件中断程序 OB40，在 Network2 中利用局部变量 OB40_MDL_ADDR 和 OB40_POINT_ADDR，在 MW10 和 MW12 中得到输入模块的起始地址和产生的中断号。



图 8-8　设置数字量输入模块的中断

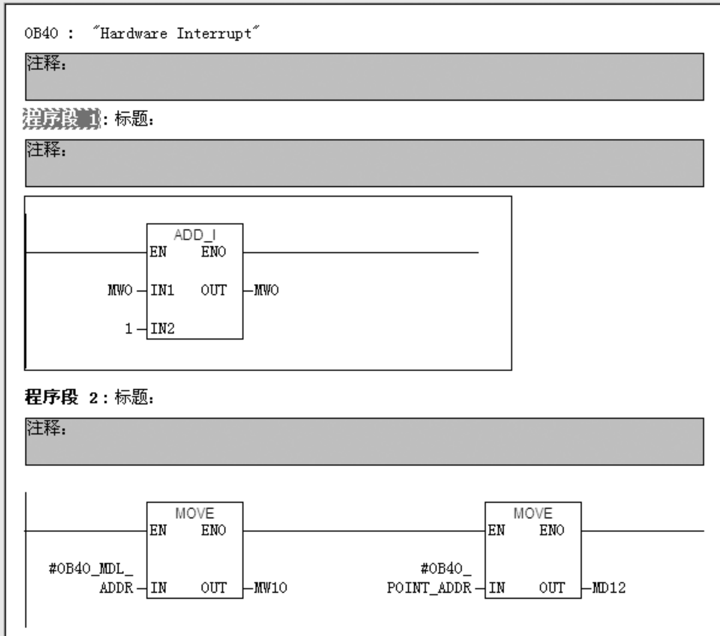


图 8-9　硬件中断程序 OB40

8.3　定期执行组织块

8.3.1　日期时间中断组织块

时间中断组织块有 OB10～OB17，共计 8 个。CPU318 只能使用 OB10 和 OB11，其余的 S7-300CPU 只能使用 OB10。日期时间中断可以在某一特定的日期和时间执行一次，也可以从设定的日期时间开始，周期性地重复执行，例如，每分钟、每小时、每天甚至每年执行一次。可以用 SFC28～SFC30 设置、取消或激活日期时间中断。系统已经在 OB10 中定义了一

些局部变量，为用户编程提供了便捷。见表 8-7。

表 8-7　OB10 的局部变量

变　　量	类　　型	描　　述
OB10_EV_CLASS	BYTE	事件等级和标识符：B#16#11 ＝ 中断处于激活状态
OB10_STRT_INFO	BYTE	B#16#11 ~ B#16#18 分别是 OB10 ~ OB17 的启动请求
OB10_PRIORITY	BYTE	分配的优先级：默认值为 2
OB10_OB_NUMBR	BYTE	OB 编号（OB10 ~ OB17）
OB10_RESERVED_1	BYTE	保留
OB10_RESERVED_2	BYTE	保留
OB10_PERIOD_EXE	WORD	OB 以指定的时间间隔执行：W#16#0000：单次 W#16#0201：每分钟一次 W#16#0401：每小时一次 W#16#1001：每天一次 W#16#1201：每周一次 W#16#1401：每月一次 W#16#1801：每年一次 W#16#2001：月末
OB10_RESERVED_3	INT	保留
OB10_RESERVED_4	INT	保留
OB10_DATE_TIME	DATE_AND_TIME	调用 OB 时的日期和时间

为了启动时间中断，用户首先必须设置时间中断的参数，然后再激活它。用户可以用组态或编程的方法来启动时间中断。

（1）设置和启动日期时间中断

有下面三种可能的启动方式。

① 基于硬件组态的自动启动时间中断，在硬件组态工具中设置和激活。在 STEP 中打开硬件组态工具，双击机架中 CPU 模块所在的行，打开设置 CPU 属性的对话框，单击“日时间中断”选项卡，设置启动日期时间中断的日期和时间，选中“激活”复选框，在“执行”列表中选择执行方式为“一次”，如图 8-10 所示。将硬件组态数据保存编译下载到 CPU 中，可以实现日期时间中断的自动启动。



图 8-10　设置和启动日期时间中断

② 用上述方法设置日期时间中断的参数，但不选择“激活”而是在用户程序中用SFC30“ACT_TINT”激活日期时间中断。

③ 可以在程序中通过调用SFC28“SET_TINT”来设置时间中断，然后通过调用SFC30“ACT_TINT”来激活它。

（2）查询日期时间中断

要想查询设置了哪些日期时间中断，以及这些中断什么时间发生，可以调用SFC31“QRY_TINT”查询日期时间中断。SFC31输入的状态字节（STATUS）见表8-8。

表8-8　SFC31输出的状态字节

位	取　值	意　义
0	0	日期时间中断以被激活
1	0	允许新的日期时间中断
2	0	日期时间中断未被激活或时间已过去
3	0	
4	0	没有装载日期时间中断组织块
5	0	日期时间中断组织块的执行没有被激活的测试功能禁止
6	0	以基准时间为日期时间中断的基础
7	1	以本地时间为日期时间中断的基础

（3）禁止和激活日期时间中断

用SFC29“CAN_TINT”取消日期时间中断，用SFC28“SET_TINT”重新设置那些被禁用的日期时间中断，SFC30“ACT_TINT”重新激活日期时间中断。SFC28的参数说明见表8-9。其他系统功能的详细使用可以查阅帮助手册。

表8-9　SFC28的参数说明

参　数	声　明	数据类型	存　储　区	描述
OB_NR	INPUT	INT	I、Q、M、D、L、常数	在时间SDT + PERIOD的倍数处启动的OB的编号（OB10～OB17）。
SDT	INPUT	DT	D、L、常数	启动日期和时间：将忽略指定的启动时间的秒和毫秒值，并将其设置为0。如果要设置每月启动时间中断OB，则只能使用日期1、2、…28号作为启动日期。
PERIOD	INPUT	WORD	I、Q、M、D、L、常数	从启动点SDT开始的周期：W#16#0000 = 一次；W#16#0201 = 每分钟；W#16#0401 = 每小时；W#16#1001 = 每日；W#16#1202 = 每周；W#16#1401 = 每月；W#16#1801 = 每年；W#16#2001 = 月末
RET_VAL	OUTPUT	INT	I、Q、M、D、L	如果在功能处于激活状态时出错，则RET_VAL的实际参数将包含错误代码。

（4）应用实例

【例8-6】 自2006-2-12的18点整开始，每分钟中断一次，每次中断使MW20自动加1。要求用I0.0的上升沿脉冲设置和启动日期时间中断OB10，用I0.1的高电平禁止日期中断OB10。

图8-11的（1）～（6）所示为主程序OB1，图8-12所示为中断程序OB10。在程序段

1 中调用系统 IEC 功能 FC3“DATE and TOD to DT”，将格式为 DATE 的日期和格式为 TOD 的时间数据合并，并且转换为 DATE_AND_TIME 格式（简称 DT）的数据。并将其暂时置于局部变量 OB1_DATE_TIME。因为在 SFC28 的输入参数 SDT（设置中断的起动起始时间）的数据类型为 DT 格式，所以必须进行数据类型的合并和转换。

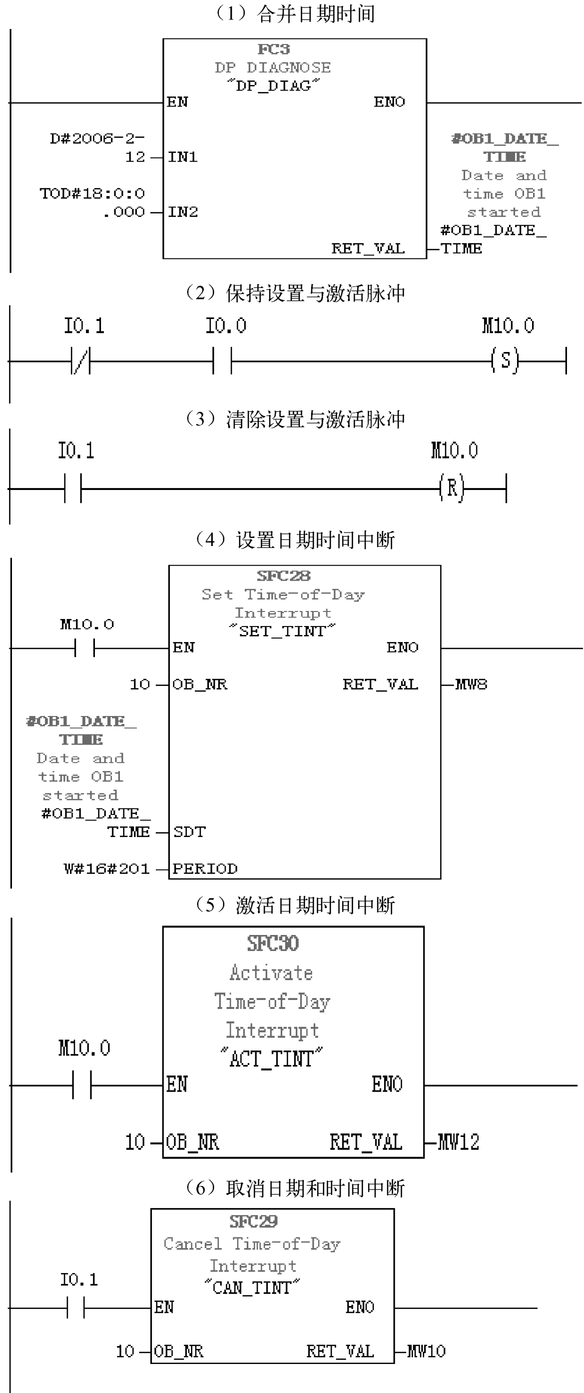


图 8-11　主程序 OB1

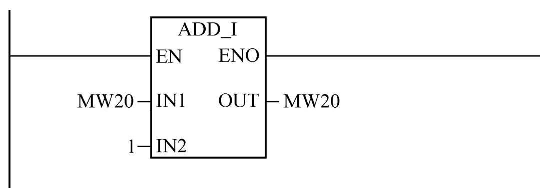


图 8-12　中断程序 OB10

8.3.2　循环中断组织块

（1）概述

STEP 7 提供了 9 个循环中断组织块（OB30 ~ OB38），它们经过一段固定的时间间隔中断用户的程序。循环中断用于按精确的时间间隔循环执行中断程序，例如周期性地执行闭环控制系统的 PID 控制程序，间隔时间从 STOP 切换到 RUN 模式时开始计算。

时间间隔不能小于 5ms。如果时间间隔过短，还没有执行完循环中断程序又开始调用它，将会产生时间错误事件，CPU 调用 OB80。如果没有创建和下载 OB80，CPU 将进入 STOP 模式下。

S7 -300 系列 PLC 中 CPU318 能使用 OB32 和 OB35，其余 S7 -300CPU 只能使用 OB35，表 8-10 所示为每个循环中断组织块默认的时间间隔和优先级，用户可以设置自己需要的时间间隔和优先级。

表 8-10　循环中断组织块默认的时间间隔和优先级

OB 号	默认时间间隔	默认的优先级
OB30	5 s	7
OB31	2 s	8
OB32	1 s	9
OB33	500 ms	10
OB34	200 ms	11
OB35	100 ms	12
OB36	50 ms	13
OB37	20 ms	14
OB38	10 ms	15

下面以 OB35 为例来说明其用法。

循环中断组织块 OB35 是按设定的时间间隔循环执行的中断程序，间隔时间从 STOP 切换到 RUN 模式时开始计时。用户可以在 OB35 中周期的调用 PID（FB41、FB42、FB43），完成 PID 调节，也可以在 OB35 中调用周期的函数据发送指令，完成数据发送功能。表 8-11 所示为 OB35 的局部变量。

表 8-11　OB35 的局部变量

变　　量	类　　型	描　　述
OB35_EV_CLASS	BYTE	时间级别和识别码：B#16#11：中断激活
OB35_STRT_INF	BYTE	B#16#30：循环中断组织块 OB 的启动请求（只有 H 型 CPU 并且明确地为其组态）B#16#31 ~ B#16#39：OB30 ~ OB38 的启动请求
OB35_PRIORITY	BYTE	分配的优先级：默认 7（OB30）~ 15（OB38）

（续）

变　　量	类　　型	描　　述
OB35_OB_NUMBR	BYTE	OB 号（30～38）
OB35_RESERVED_1	BYTE	保留
OB35_RESERVED_2	BYTE	保留
OB35_PHASE_OFFSET	WORD	相位偏移量［ms］
OB35_EXC_FREQ	INT	时间间隔，以 ms 计
OB35_DATE_TIME	DATE_AND_TIME	OB 调用的日期和时间

如果两个 OB 的时间间隔成整数倍，不同的循环中断 OB 可以同时请求中断，造成处理循环中断程序超过指定的循环时间。为了避免出现这样的错误，用户可以定义一个相位偏移。相位偏移用于循环时间间隔到达时，延时一定的时间后再执行循环中断，相位偏移时要小于循环的时间间隔。比如可以设置 OB37 和 OB38 的时间间隔分别为 10 ms 和 20 ms，它们的相位偏移分别为 0 ms 和 3 ms，则 OB37 分别在 $t=10$ ms，20 ms，…时产生中断，而 OB38 分别在 $t=23$ ms，43 ms，…时产生中断。

（2）应用方法

首先可以在 STEP 7 中查看可支持的循环中断 OB。具体方法：在 STEP 7 的硬件的组态窗口中，双击项目机架上的 CPU 所在的行，打开 CPU 属性对话框，单击“循环中断”选项页，设置框就显示当前 CPU 可以使用的循环中断块，可以设置循环中断，如图 8-13 所示。用户可以在“优先级”编辑框中设置当前循环 OB 的优先级，在“执行”编辑框中可改变默认的时间间隔，范围是 0～60000 ms。

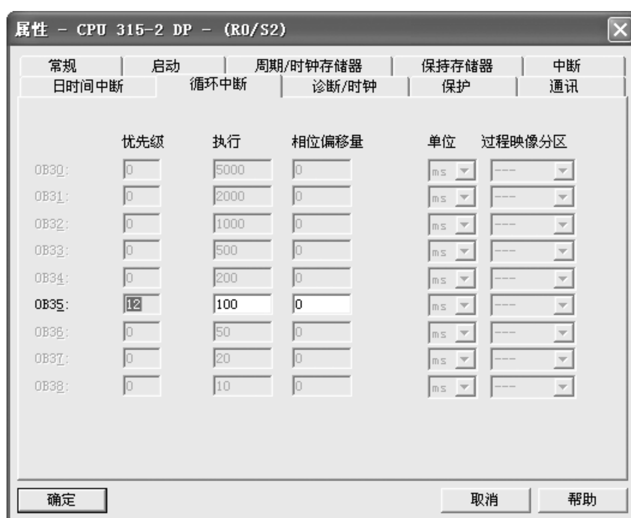


图 8-13　设置循环时间

循环中断组织块每一次运行时间一定要短于中断的间隔，如果一个循环中断组织块没有执行完，循环中断时间到，又要求循环中断组织块运行，则时间故障组织块 OB80 启动。反复的循环中断将导致故障程序的运行。

与 OB20 使用方法不同的是，系统虽然没有提供专用的激活和禁止循环中断 SFC，但可以运用 SFC39～42SFC 取消、延时和再次使能循环中断。SFC40“EN_INT”是用于激活新的中断和异步错误的系统功能，其参数 MODE 为 0 时激活所有的中断和异步错误；为 1 时激活部分中断和异步错误，为 2 时激活指定的 OB 编号对应的中断和异步错误。SFC39“DIS_INT”是禁

止新的中断和异步错误的系统功能，其参数 MODE 为 2 时禁止指定的 OB 编号对应的中断和异步错误，MODE 参数有多种选择，其他可查阅系统手册《SIMATIC S7 －300 的系统软件和标准功能》。以上两个 SFC 的 MODE 参数必须用十六进制来设置。SFC39 ~ SFC42 的参数说明见表 8-12。

表 8-12　SFC39 ~ SFC42 的参数说明

参数	声明	数据类型	存 储 区 域	参 数 说 明
OB_NR	INPUT	INT	I、Q、M、D、L 常数	OB 号（OB30 ~ OB38）
MODE	INPUT	BYTE	I、Q、M、D、L 常数	定义被禁止的中断和异步故障
RET_VAL	OUTPUT	INT	I、Q、M、D、L	如故障发生，返回值包含故障代码 在 SFC41 中：延迟的编号（等于 SFC41 调用的编号） 在 SFC42 中：激活报警中断调用 SFC 的次数或者故障信息

（3）应用实例

【例 8-7】 每 3 分钟中断一次，每次中断使 MW0 自动加 1。要求用 I0. 1 的上升沿脉冲设置和启动循环中断 OB35，用 I0. 1 的高电平禁止循环中断 OB35。主程序 OB1 如图 8-14 的（1）～（4）所示，本例中循环中断程序与图 8-12 相同。要使用循环中断，首先在图 8-13 中设置循环间隔时间为 3000 ms，表示每 3s 调用 OB35 一次。

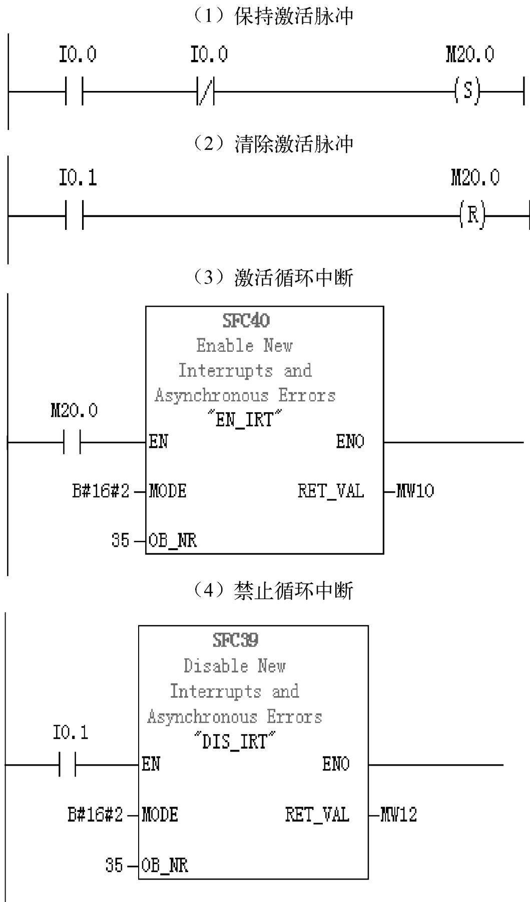


图 8-14　主程序 OB1

在OB1的（3）和（4）中MODE为B#16#2分别表示激活指定的OB35所对应的循环中断和禁止新的循环中断。程序下载保存后就可以下载到实际的PLC或PLCSIM仿真软件中了，可以在STEP 7的变量表中进行程序监控。

8.4　错误处理组织块

8.4.1　错误处理概述

西门子的大中型PLC具有很强的错误和故障的检测及处理能力。当CPU检测到某种错误后，操作系统将自动调用相应的组织块，用户可以事先在对应的组织块中编写程序，就可以对发生的错误或故障采取相应的措施。如用户没有建立相应的组织块，如果错误或故障后，CPU将转为STOP模式。如果下载空的OB，即使出现错误时CPU也不会进入STOP模式。

像所有的组织块一样，错误处理组织块中包含了关于调用原因的附加信息。操作系统将这些信息记录在该组织块的局部变量中，用户可以在程序中对它们进行访问，以便于故障诊断。当CPU不支持某些错误组织块时，相应的错误信息就不会记录在OB中。

S7－300 PLC有很强的错误检测和处理能力。操作系统可以检测出下列错误：不正确的CPU功能、操作系统执行中的错误、用户程序的错误和I/O中的错误。

操作系统检测到一个异步错误时，将启动相应的组织块（见表8-13）。异步错误OB具有最高等级的优先级，如果当前正在执行的OB的优先级低于26，异步错误OB的优先级为26，如果当前正在执行的OB的优先级为27（启动组织块），异步错误OB的优先级为28，其他OB不能中断它们。如果同时有多个相同优先级的异步错误OB出现，将按出现的顺序处理它们。利用系统功能（SFC），用户可以屏蔽、延迟或禁止各种OB的启动事件。

表8-13　错误处理组织块类型和优先级

OB号	错误类型	优先级
OB70	I/O冗余错误（仅H系列CPU）	25
OB72	CPU冗余错误（仅H系列CPU）	28
OB73	通信冗余错误（仅H系列CPU）	25
OB80	时间错误	26
OB81	电源故障	26/28
OB82	诊断中断	
OB83	插入/拔出模块中断	
OB84	CPU硬件故障	26/28
OB85	优先级错误	
OB86	机架故障或分布式I/O的站故障	
OB87	通信错误	
OB121	编程错误	错误的OB的优先级
OB122	I/O访问错误	

8.4.2　错误的分类

被 S7 CPU 检测到并且用户可以通过组织块对其进行处理的错误分为两个基础类。

① 异步错误是与 PLC 的硬件或操作系统密切相关的错误，与用户程序的执行无关。异步错误的后果一般比较严重。异步错误对应的组织块为 OB70～OB73 和 OB80～OB87，具有最高级的优先级。

② 同步错误是与程序执行有关的错误，OB121 和 OB122 用于处理同步错误，它们的优先级与出现错误时被中断的优先级相同，即同步错误 OB 的程序可以访问块被中断时累加器和状态寄存器的内容。对错误进行适当处理后，可以将处理结果返回被中断的块。

8.4.3　同步错误组织块

所谓同步错误，是在程序执行过程中，在程序的某个特定位置上发生的错误。发生这种错误时，系统调用对应的错误处理组织块。同步错误组织块具有与当前 OB 相同的优先等级，因此这时同步组织块里的程序总是被执行。同步错误组织块包括 OB121（编程错误）和 OB122（访问错误）。

OB121：当程序发生诸如 BCD 转换错误、调用一个不存在的块、访问数据块中没有经过编辑的地址等错误时，系统调用 OB121。

OB122：当程序发生诸如访问一个不存在的外设地址（PIW、PQW 等）、访问的模块有故障等错误时，系统调用 OB122。

发生同步/异步错误，系统调用相应的错误组织块，如果该错误组织块（哪怕是空的）已经下载到工作存储区，则系统在执行该错误组织块里的程序后，返回当前 OB 继续循环扫描，这时 CPU 面板上的红灯亮，但 CPU 仍然在 RUN 状态。如果系统调用相应的错误组织块时，找不到这个组织块（没有下载），则 CPU 转为 STOP 状态，CPU 面板上的 SF 红灯亮，尽管开关仍在 RUN 位置，但 STOP 橙色灯亮。同时，出错信息会被记录在诊断缓冲区。

掌握了错误 OB 的调用条件和系统处理错误 OB 的方式后，用户可以很好地利用这些错误 OB。例如：

① 为了避免发生某种错误时 CPU 进入停机状态，可以下载一个相应的空 OB。

② 可以在相应的错误 OB 中编程实现希望的响应。

③ 错误组织块的变量声明表中，有许多有用的信息（如出错原因、地址等），可以在错误组织块中编程提取这些信息，作为处理根据。

注：每个组织块的详细情况请参阅 STEP 7 在线帮助。

8.4.4　冗余错误组织块

I/O 冗余错误 OB（OB70）仅可与 H CPU 结合使用。当在 PROFIBUS DP 上存在冗余丢失时（例如，激活 DP 主站发生总线故障，或 DP 从站的接口模块出错），或者当具有连接 I/O 的 DP 从站的激活 DP 主站发生改变时，H CPU 的操作系统将调用 OB70。

如果发生了启动事件而 OB70 没有编程，则 CPU 不会转为 STOP 模式。如果装载了 OB70 且 H 系统处于冗余模式，则在两个 CPU 上执行 OB70。H 系统仍处于冗余模式。

表 8-14 给出了 I/O 冗余错误 OB 的临时（TEMP）变量。所选变量名为 OB70 的默认名称。

表 8-14　I/O 冗余错误 OB 的临时变量

变　　量	类　　型	描　　述
OB70_EV_CLASS	BYTE	事件等级和 ID：B#16#72 离开事件 B#16#73：进入事件
OB70_FLT_ID	BYTE	错误代码（可能值：B#16#A2、B#16#A3）
OB70_PRIORITY	BYTE	优先级，可通过 STEP 7 分配（硬件组态）
OB70_OB_NUMBR	BYTE	OB 编号（70）
OB70_RESERVED_1	WORD	保留
OB70_INFO_1	WORD	取决于错误代码
OB70_INFO_2	WORD	取决于错误代码
OB70_INFO_3	WORD	取决于错误代码
OB70_DATE_TIME	DATE_AND_TIME	调用 OB 时的日期时间

习题

8-1　填空。

（1）S7 -300 PLC 在启动时调用________。

（2）CPU 检测到故障或错误时，如果没有下载对应的错误处理 OB，CPU 将进入________。

（3）异步错误是与 PLC 的________或________有关的错误。

（4）同步错误是与________有关的错误，OB ________和 OB ________用于处理同步错误。

（5）OB 具有不同的优先级，优先级的范围是________，其中________优先级最低，________优先级最高。

8-2　组织块（OB）有哪几类？各自有什么作用？

8-3　简述异步组织块的作用。

8-4　利用循环组织块 OB35 建立振荡电路，周期为 2s。

8-5　要求每 600 s 在 OB35 中将 MW0 加 1，在 I0.0 的上升沿停止调用 OB35，在 I0.1 的上降沿允许调用 OB35。生成项目，组态硬件，编写程序，用 PLCSIM 调试程序。

8-6　在主程序 OB1 中实现以下功能。

（1）在 I0.0 的上升沿用 SFC32 启动延时中断 OB20，10 s 后 OB20 被调用，在 OB20 中将 Q4.0 置位，并立即输出。

（2）在延时过程中如果 I0.1 由 0 变为 1，在 OB1 中用 SFC33 取消延时中断，OB20 不会再被调用。

（3）I0.2 由 0 变为 1 时 Q4.0 被复位。

8-7　分析图 8-15 所示程序的功能。

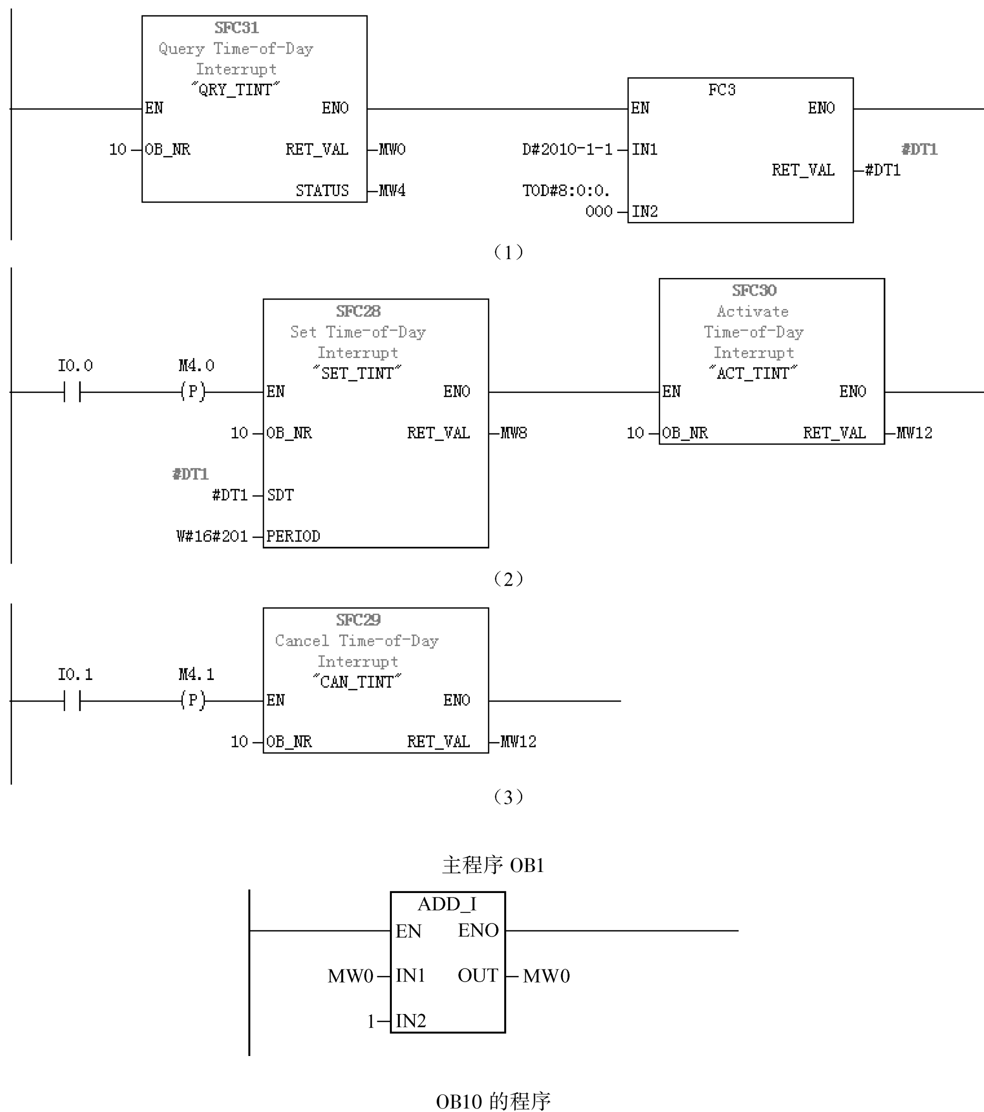


图 8-15　题 8-7 图

第 9 章　S7 - 300/400 PLC 在模拟量闭环控制中的应用

9.1　PID 控制器

9.1.1　PID 控制的基本原理

PID 控制是由 P（比例）、I（积分）、D（微分）三种运算以和的形式组合在一起的一种控制方式。PID 控制使用灵活，用途广泛，即使在控制理论与技术高速发展的今天，在工业过程控制中，绝大多数控制回路都具有 PID 控制结构，尤其是在被控对象数学模型不是很清楚的情况下，使用 PID 控制具有比较大的优势。其算式如下

$$p = K_p\left(e + \frac{1}{T_i}\int e\mathrm{d}t + T_d\frac{\mathrm{d}e}{\mathrm{d}t}\right) \tag{9-1}$$

式中　p—调节器的输出信号；

e—调节器的输入信号，是测量值与给定值之差；

K_p—调节器的比例系数；

T_i—调节器的积分时间常数；

T_d—调节器的微分时间常数。

PID 控制器对偏差信号进行比例、积分、微分运算，其结果输出给执行器，其在控制系统中的位置图如图 9-1 所示。

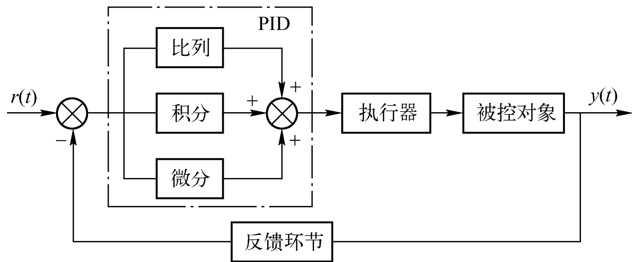


图 9-1　PID 控制器在系统中位置

9.1.2　PID 控制器的数字化

由于采样周期 T_s 相对于信号变化周期是很小的，因此，可以用矩形面积法计算积分，用后向差分法代替微分，则式（9-1）中的积分项与微分项可分别近似表示为

$$\int e\mathrm{d}t \approx \sum_{i=0}^{n} e_i\Delta t = T_s\sum_{i=0}^{n} e_i \tag{9-2}$$

$$\frac{\mathrm{d}e}{\mathrm{d}t} \approx \frac{e_n - e_{n-1}}{T_s} \tag{9-3}$$

式（9－1）便成了离散 PID 算式

$$p_n = K_p\left[e_n + \frac{T_s}{T_i}\sum_{i=0}^{n} e_i + \frac{T_d}{T_s}(e_n - e_{n-1})\right] \tag{9-4}$$

式中 Δt—采样周期，与 T_s 相等；

p_n—第 n 次采样时调节器的输出；

e_n—第 n 次采样的偏差值；

n—采样序号。

式（9－4）称作数字 PID 的位置式算式。还有一种增量式算式，其求法如下。

由式（9－4）可求得第（$n-1$）次采样的输出表达式

$$p_{n-1} = K_p\left[e_{n-1} + \frac{T_s}{T_i}\sum_{i=0}^{n-1} e_i + \frac{T_d}{T_s}(e_{n-1} - e_{n-2})\right] \tag{9-5}$$

由式（9－4）减去式（9－5），可得 PID 控制器增量式算式表达式

$$\Delta p_n = (K_p + K_i + K_d)e_n - (K_p + 2K_d)e_{n-1} + K_d e_{n-2} \tag{9-6}$$

式中 K_i—PID 控制算式的积分系数，$K_i = K_p\dfrac{T_s}{T_i}$；

K_d—PID 控制算式的微分系数，$K_d = K_p\dfrac{T_d}{T_s}$。

增量式算式中，PID 控制的输出信号是本次 PID 计算结果与上次 PID 输出值之和。

9.1.3　PID 控制器参数与系统性能关系

比例部分与偏差成比例关系，它的控制作用与误差同步，在误差出现时，比例控制能立即起作用，使被控量朝着误差减少的方向变化，但单纯的比例控制存在稳态误差。比例系数 K_p 对系统的影响反映在系统的稳态误差与稳定性上。增大 K_p 可以增加系统开环增益，减少系统的稳态误差，但若 K_p 增加过量会导致超调量增大，振荡次数增加，调节时间变长，动态性能变差。若 K_p 过大可能会导致系统不稳定。因此，单纯的比例控制很难兼顾控制过程的动态性能与稳态性能。

积分部分对偏差进行积分，只要偏差存在，就会随着时间累积积分量，使得控制向减少误差的方向变化。积分部分与当前偏差值和过去的历次偏差值之和成正比，因此，具有滞后性。积分作用能消除系统的误差，提高控制精度，理论上，只要有积分部分存在，误差最终会完全消除。积分时间常数 T_i 对积分作用影响大，如果 T_i 太小，积分部分迅速累积，积分作用强。若 T_i 过小，会使得系统震荡，超调量增大，调节时间过长，甚至可能会导致积分饱和，使得系统不稳定；如果积分时间常数太大，积分作用不明显，系统消除偏差的时间变长，因此，T_i 要取值适宜。

微分部分与偏差的变化率成正比例关系，只有误差随时间变化时，它才起作用。微分部分具有超前和预测的特性，根据被控量变化的趋势，微分部分能提前采取措施，以减少超调量。微分时间常数 T_d 与微分作用的强弱成正比，T_d 越大，微分作用越强。若 T_d 过大，会对偏差过分抑制，反而会加剧系统的震荡，使超调量增大。T_d 太小，则微分作用不强，若将 T_d 设置为 0，则微分不起作用。因此，T_d 取值也要适宜。

当偏差很小的时候，比例部分与微分部分的作用非常小，可以忽略不计，积分部分仍在

进行累积，PID 输出值主要是积分量。当系统处于稳定状态时，偏差恒为零，比例部分与微分部分均为零，积分部分也不再变化，此时，PID 控制器的输出就是积分量。

根据比例、积分、微分各部分的作用特点，可以知道，比例、积分与微分一般不单独使用，常用的形式有 PI、PID。这样不仅克服了单纯的比例调节存在稳态误差的缺点，又避免了单纯的积分调节响应慢、动态性能不好的缺点。

9.2 连续 PID 控制模块 FB41

FB41“CONT_C”（连续控制器）的输出为连续变量。其采用的算法是 PID 算法。FB41 与 FB43 结合使用，可以产生脉冲宽度可调的输出信号，来控制比例执行机构。FB41 的结构框图如图 9-2 所示。

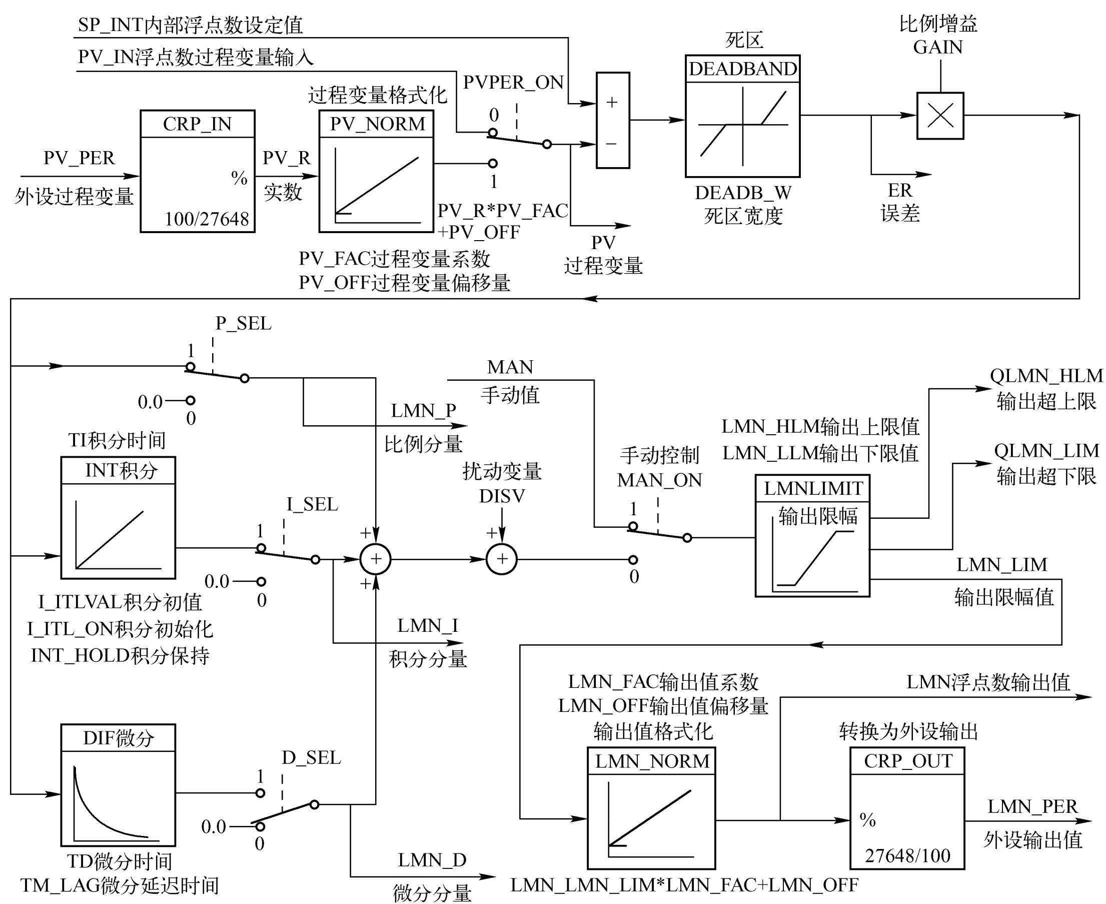


图 9-2　FB41 功能框图

9.2.1　设定值与过程变量的处理

1. 设定值的输入

在 FB41 内部，PID 控制器的设定值 SP_INT、过程变量输入 PV_IN 是浮点数。过程变量可以采用两种方式输入。

① BOOL 型参数 PVPER_ON 为 0 状态时，通过 PV_IN 输入以百分数为单位的浮点数格式的过程变量参与到 FB41 中进行运算。

② BOOL 型参数 PVPER_ON 为 1 状态时，通过 PV_PER 输入模拟量输入模块输出的数字值作为过程变量参与到 FB41 中进行运算。

2. 外部设备过程变量转换为浮点数

图 9-2 中的 CRP_IN 方框将 0 ~ 27648 或 ±27648（对应模拟量输入的满量程）的外部设备过程变量 PV_PER，转换为 0 ~ 100% 或 ±100% 的浮点数格式的百分数，CPR_IN 的输出 PV_R 用下式计算：

$$PV_R = PV_PER \times 100/27648(\%)$$

3. 外设变量的格式化

PV_NORM（外设变量格式化）方框通过用下面的公式将 CRP_IN 方框的输出 PV_R 格式化：

$$PV_NORM 的输出 = PV_R \times PV_FAC + PV_OFF$$

式中，PV_FAC 为过程变量的系数，默认值为 1.0；PV_OFF 为过程变量的偏移量，默认值为 0.0。PV_FAC 和 PV_OFF 用于调节外设输入过程变量的范围。

4. 设定值与过程变量单位的一致性

设定值 SP_INT 与过程变量应该有相同的单位。如果过程变量是通过 PV_IN 输入，若 PV_IN 是“工程量值”，例如 10MPa 或 200℃等，则 SP_INT 也必须是“工程量值”；若 PV_IN 是“工程量值百分比”，例如 30% 等，则 SP_INT 也必须是“工程量值百分比”。如果过程变量是通过 PV_PER 外围设备输入的，则 SP_INT 只能使用“ -100.0% ~100.0%”之间的数据作为设定值。例如若要将 0 ~ 10 MPa 中的 8 MPa 作为设定值，则 SP_INT 设为 80.0% 或 0.8。

9.2.2 控制器算法

1. 误差的计算与死区特性

SP_INT（内部设定值）是以百分数为单位的浮点数设定值。用 SP_INT 减去浮点数格式的过程变量 PV（反馈值），得到误差值。

DEAD BAND（死区）可以设置死区宽度。当系统控制精度可以允许存在一定误差时，可以将允许的最大误差设为 DEADB_W（死区宽度）的值。当误差的绝对值小于等于 DEADB_W 时，此时 PID 的输入量是 0，而不是误差值，PID 控制器不进行新的运算，PID 的输出量与上一次的输出量相同。当误差的绝对值大于 DEADB_W 时，PID 进行正常的运算。对于允许存在一定误差的控制过程，设置 DEAD BAND 可以减少 PID 的运算，从而减少执行机构的频繁动作造成不必要的机械磨损。

2. 控制器结构设置

FB41 采用位置式 PID 算法，P_SEL、I_SEL、D_SEL 是 BOOL 型参数，分别控制比例、积分、微分是否起作用。其值设为 1 时，启用对应的控制作用，设为 0 时，则禁止相应的控制作用。在实际的 PID 控制方式中，很少用单独的积分或微分控制，默认的控制方式为 PI 控制，根据实际需要，可以组态 P、PI、PD、PID 几种方式。

LMN_P、LMN_I、LMN_D 分别是 PID 控制器输出量中的比例分量、积分分量、微分分量，供调试时用。

图 9-2 中的 GAIN 为比例系数，对应式（9-1）中的 K_p，TI 与 TD 分别为积分时间常数与

微分时间常数，对应于式（9－1）中的 T_i 和 T_d。TM_LAG 为微分延迟时间，FB41 帮助文件中建议其设置为 TD/5。DISV 是扰动变量，其默认值是 0，通过引入 DISV 可以实现前馈控制。

FB41 有一个初始化程序，在参数 COM_RST（完全重新启动）为 1 时，该程序执行。初始化过程中，如果 BOOL 型参数 I_ITL_ON 为 1，表示积分作用要初始化，需要给 I_ ITLVAL 输入一个值作为积分部分的初始值。INT_HOLD 为 1 时，积分操作保持不变，积分输出被冻结，一般 INT_HOLD 设置为 0。

3. 手动模式

FB41 有两种工作模式，即自动模式与手动模式。在自动模式情况下，PID 的输出是对偏差进行 PID 运算后的结果，手动模式情况下，PID 的输出由人工给定。模式切换由 BOOL 型参数 MAN_ON 控制，其值设置为 1 时，FB41 工作在手动模式下，此时 PID 控制器的输出值由 MAN 值确定；当 MAN_ON 值设置为 0 时，FB41 工作在自动模式下。

在手动模式下，控制器输出中的积分分量被自动设置为 LMN－LMN_P－DISV，微分分量被自动设置为 0。这样可以保证从手动模式切换到自动模式时，PID 控制器在切换瞬间前后的输出值 LMN 不发生突变，有利于减少对执行器的冲击。

9.2.3 输出值的处理

1. 输出限幅

LMNLIMIT（输出量限幅）用于将控制器输出值 LMN 限幅。LMNLIMIT 的输入量超出控制器输出值的上限值 LMN_HLM 时，LMNLIMIT 的输出值 LMN_LIM 为 LMN_HLM，BOOL 型参数 QLMN_HLM（输出超出上限）输出为 1；LMNLIMIT 的输入量小于下限值 LMN_LLM 时，LMNLIMIT 的输出值 LMN_LIM 为 LMN_LLM ，BOOL 型参数 QLMN_LLM（输出低于下限）输出为 1。LMN_HLM 与 LMN_LLM 的默认值分别为 100.0% 和 0.0% 。

2. 输出值格式化

LMN_NORM（输出量格式化）功能是将 LMN_LIM 格式化：

$$\text{LMN} = \text{LMN_LIM} \times \text{LMN_FAC} + \text{LMN_OFF}$$

式中，LMN 是控制器的输出，数据类型为浮点数；LMN_FAC 是输出系数，默认值为 1.0；LMN_OFF 是输出值偏移量，默认值为 0.0。LMN_FAC 与 LMN_OFF 用于调节控制器输出值的范围。

3. 转换为外设输出

CPR_OUT（转换为外设输出）的功能是将 LMN（0～100% 或 ±100% 的浮点数格式的百分数）转换为外部设备格式的变量 LMN_PER（0～27648 或 ±27648 的整数），转换公式为

$$\text{LMN_PER} = \text{LMN} \times 27648/100$$

9.3 脉冲发生器模块 FB43

9.3.1 脉冲发生器的功能与结构

1. 脉冲发生器的功能

FB43“PULSEGEN”（脉冲发生器）可以输出宽度可调的脉冲信号，用于控制比例执行

机构。其一般与连续 PID 控制器 FB41 配合使用，FB41 的输出 LMN 作为 FB43 的输入 INV。FB43 可以构建宽度可调的两步（two step）或三步（three step）PID 控制器。FB43 与 FB41 配合使用输出脉冲的闭环控制系统框图如图 9-3 所示。

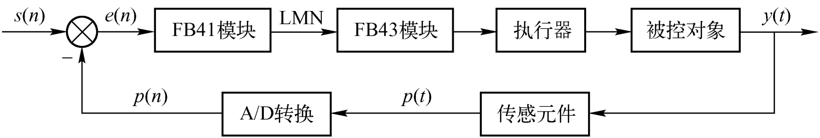


图 9-3　FB41 与 FB43 配合使用构成闭环控制系统框图

FB43 将输入量 INV（也即 FB41 的输出 LMN）转换为在一恒定时间内具有一定宽度的脉冲，脉冲宽度与 INV 成正比例关系。恒定时间（也即周期时间）是由参数 PER_TM 设置，PER_TM 应与 FB41 的采样时间 CYCLE 相同。P_B_TM 称为 FB43 的处理周期，即 FB43 的调用周期，PER_TM 是 P_B_TM 的若干倍。在一个 PER_TM 内包含的 P_B_TM 越多，则脉宽控制的精度越高。假设一个 PER_TM 含有 10 个 P_B_TM，即在一个 PER_TM 内调用 10 次 FB43，如果 INV 的值为 80，则在前 8 次调用 FB43 时，FB43 的 QPOS_P 均为 1，剩下两次调用 FB43 时，QPOS_P 为 0，即此时脉冲 QPOS_P 的占空比为 8∶10。其示意图如图 9-4 所示。

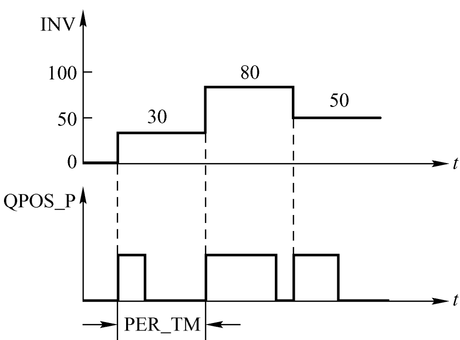


图 9-4　INV 值与宽度可调脉冲对应关系图

另外，可以通过 SYN_ON（启用同步）参数来设置同步功能。如果 SYN_ON 设为 TRUE，当 INV 发生了变化，只要 FB43 不在原输出脉冲的第一个周期或最后两个调用周期，将进行同步，即 FB43 根据新的 INV 值重新计算脉冲宽度，并在下一个恒定周期中输出一个新的宽度与 INV 值相对应的脉冲。

2. 运行模式的参数设置

根据脉冲发生器所赋值的参数，可以组态成具有一个三步输出，或者一个两向两步输出或单向两步输出。在手动模式下，三步控制或两步控制的输出由手动输入信号 POS_P_ON 和 NEG_P_ON 来控制，与输入量 INV 无关。表 9-1 给出了各种运行模式的参数设置。表 9-2 给出了手动模式下输入与输出的关系。

表 9-1　运行模式的参数设置

运 行 模 式	MAN_ON	STEP3_ON	ST2BI_ON
三步控制	FALSE	TRUE	ANY
双向两步控制（-100% ~100%）	FALSE	FALSE	TRUE

（续）

运 行 模 式	MAN_ON	STEP3_ON	ST2BI_ON
单向两步控制（0% ~100%）	FALSE	FALSE	FALSE
手动模式	TRUE	ANY	ANY

表 9-2　手动模式下输入与输出的关系

控 制 模 式	手 动 输 入		开关量输出	
	POS_ P_ ON	NEG_ P_ ON	QPOS_ P	QNEG_ P
三步控制	FALSE	FALSE	FALSE	FALSE
	TRUE	FALSE	TRUE	FALSE
	FALSE	TRUE	FALSE	TRUE
	TRUE	TRUE	FALSE	FALSE
两步控制	FALSE	ANY	FALSE	TRUE
	TRUE	ANY	TRUE	FALSE

在输入参数 COM_RST 为 1 时，FB43 进行初始化，所有输出信号被设置为 0。

9.3.2　两步控制器

两步控制器用产生 QPOS_P 与 QNEG_P 两种相反状态的脉冲控制信号，例如 QPOS_P 为 1 时，QNEG_P 为 0。如果执行器需要逻辑状态相反的开关量信号，可以用 QPOS_P 与 QNEG_P 给予提供。FB43 的脉冲输出 QPOS_P 与 INV 的值成正比例关系。根据控制值 INV 的极性，两步控制器分为双极性控制器与单极性控制器，图 9-5 与图 9-6 分别给出了两种情况下 FB43 的输出 QPOS_P 与 INV 的关系。

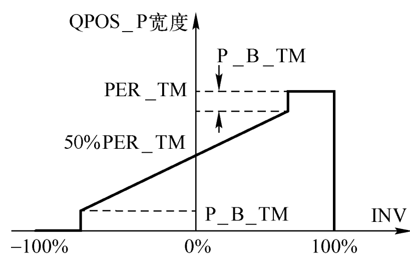


图 9-5　双极性下 QPOS_P 与 INV 的关系

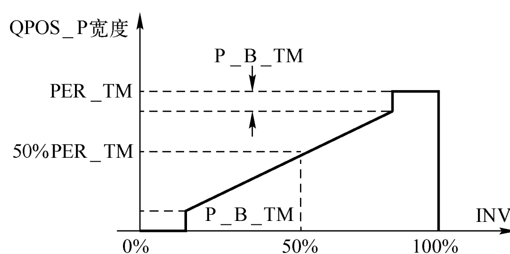


图 9-6　单极性下 QPOS_P 与 INV 的关系

从图 9-5 与图 9-6 可知，FB43 的输出脉冲 QPOS_P 的宽度与 INV 的值大部分情况下成正比例关系。当 INV 比较小，计算出来的 QPOS_P 宽度小于 P_B_TM 时，则 QPOS_P 为 0，不输出脉冲。当 INV 比较大，计算出来的 QPOS_P 宽度与 PER_TM 之差小于 P_B_TM 时，则 QPOS_P 的脉冲宽度等于 PER_TM，即在整个恒定周期内，QPOS_P 一直为正。

9.3.3　三步控制器

三步控制器输出的脉冲信号 QPOS_P 与 QNEG_P 没有必然联系，FB43 根据输入量 INV

的大小来计算 QPOS_P 与 QNEG_P 的宽度。当 INV 大于 0 时，QPOS_P 输出相应宽度的脉冲，QNEG_P 没有输出；当 INV 小于 0 时，QNEG_P 输出相应宽度的脉冲，QPOS_P 没有输出。用三步控制器来控制电动调节阀的开度时，可用 QPOS_P 使调节阀的伺服电动机正转，阀的开度增大，可用 QNEG_P 使调节阀的伺服电动机反转，阀的开度减少。

另外，可以通过改变 RATIOFAC（比率系数）来改变 QPOS_P 脉冲宽度与 QNEG_P 脉冲宽度之比。例如可以将不同的时间常数用于热处理的电加热执行机构与水冷却执行机构。

比率系数小于 1 时脉冲宽度的计算公式为

$$\text{QPOS_P 脉冲宽度} = \text{INV} \times \text{PER_TM}/100$$

$$\text{QNEG_P 脉冲宽度} = \text{INV} \times \text{PER_TM} \times \text{RATIOFAC}/100$$

比率系数大于 1 时脉冲宽度的计算公式为

$$\text{QPOS_P 脉冲宽度} = \text{INV} \times \text{PER_TM}/(100 \times \text{RATIOFAC})$$

$$\text{QNEG_P 脉冲宽度} = \text{INV} \times \text{PER_TM}/100$$

显然，比率系数小于 1 时，QNEG_P 脉冲宽度将会减少。比率系数大于 1 时，QPOS_P 脉冲宽度将会减少。图 9-7 与图 9-8 分别是比率系数为 1 时的三步控制器的对称的 INV 与脉冲输出关系曲线和脉冲输出情况。图 9-9 与图 9-10 分别是比率系数小于 1 时和比率系数大于 1 时的三步控制器的非对称的 INV 与脉冲输出关系曲线。

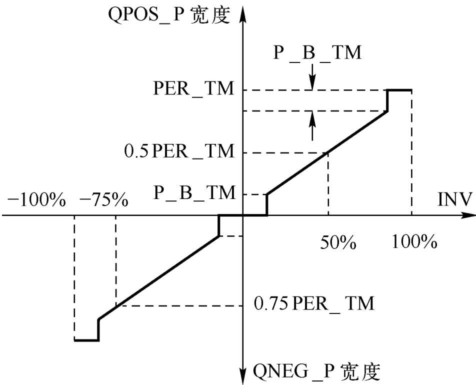


图 9-7　INV 与脉冲输出关系曲线（RATIOFAC = 1）

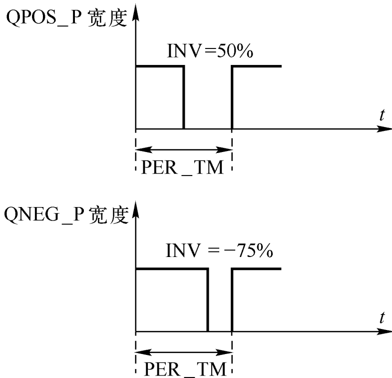


图 9-8　脉冲输出示意图

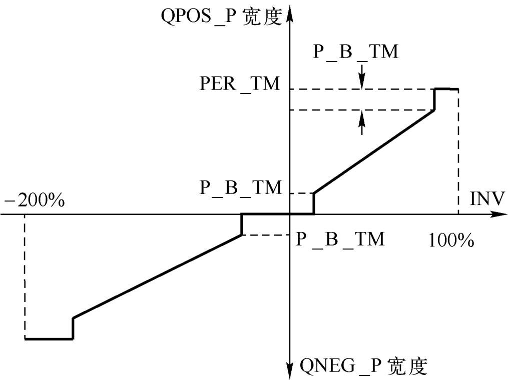


图 9-9　INV 与脉冲输出关系曲线（RATIOFAC = 0.5）

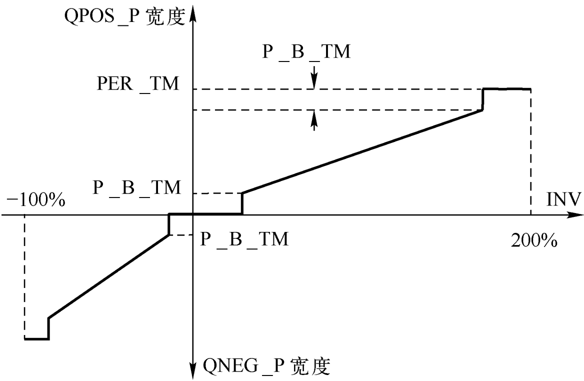


图 9-10　INV 与脉冲输出关系曲线（RATIOFAC = 2）

9.4 闭环控制示例

9.4.1 系统简介

在生产过程控制领域，对电机调速是经常遇到的问题，例如，在一些给料的控制过程中，就是通过调节驱动输送带电机的速度来实现的。本例以 PLC 为核心组成一个闭环控制系统实现电机的调速。其系统框图如图 9-11 所示。

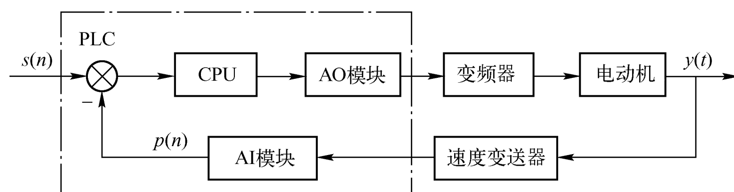


图 9-11 电机调速系统框图

工作原理：测速变送器检测电机转速，输出 4～20 mA 的电流，经模拟量输入模块 A/D 转换后送入 CPU 与设定值比较，差值进行 PID 运算，运算结果经输出量模块 D/A 转换后输出一个 0～10 V 之间的某个电压，对应 0～50 Hz 之间的某个频率，驱动电机转动。只要实际转速与设定值不一致，PID 就会计算不断输出新的值调整电动机电源频率，改变电机速度，直到两者相等为止。

9.4.2 硬件选型及信号类型设置

系统中电机选用三相异步电机，型号为 YS6322，额定功率 250 W，额定转速 2800 r/min，采用星型接法，额定电压为 220 V，额定电流为 1.17 A。变频器选用型号为 VFD007B43A 的台达变频器，额定电压 380～460 V，适配电机功率为 0.75 kW。速度变送器由 SZCB-01 型磁性转速传感器与 SZC-06 型转速变送单元组成，磁性转速传感器输出与电机转速成正比例的电压脉冲信号，经变送单元处理后输出 4～20 mA 电流，测量范围 0～9999 r/min。

PLC 电源模块选用 PS 307 2A，型号：6ES7 307-1BA00-0AA0；CPU 选用 CPU 315，型号：6ES7 315-1AF03-0AB0/V1.2；模拟量输入模块选用 SM331 AI2×12 Bit，型号：6ES7 331-7KB00-0AB0；模拟量输出模块选用 SM332 AO2×12 Bit，型号：6ES7 332-5HB00-0AB0。

本例中模拟量输入模块具有两个通道，信号类型选用 4 线制 4～20 mA 电流信号，值得注意的是要将安装在模拟量输入模块侧面的量程卡上的“C”字母对着模板上的箭头。模块输入信号设置如图 9-12 所示。

本例中模拟量输出模块具有两个通道，输出信号类型选用 0～10 V 的电压信号，用“E”表示。模块输出信号设置如图 9-13 所示。



图 9-12　AI 模块输入信号设置



图 9-13　AO 模块输出信号设置

9.4.3　程序设计

1. 初始化程序

在 OB100 中编写如下程序，在 CPU 上电或操作模式更改时，在循环程序执行之前，对 FB41 进行初始化，对 MW0、MW10、MD20 清零。程序如图 9-14 所示。

2. 主程序

在 OB35 中编写如下程序，中断时间为 150 ms，其中程序段 1 是电机实际转速采集程序，本例中速度传感器测速范围为 0 ~ 9999 r/min，所以参数 HI_LIM 和 LO_LIM 分别设为 9999.0 与 0.0；传感器信号接入模拟量输入模块第一通道，其地址为 256，所以 FC105 模块中参数 IN 接 PIW256。输出模块选用第一通道输出，其地址为 272，所以 FB41 模块 LMN_PER 接 PQW272。程序如图 9-15 所示。

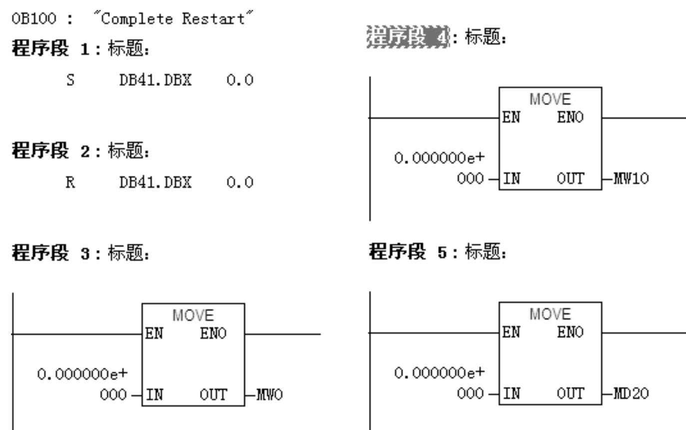


图 9–14　初始化程序

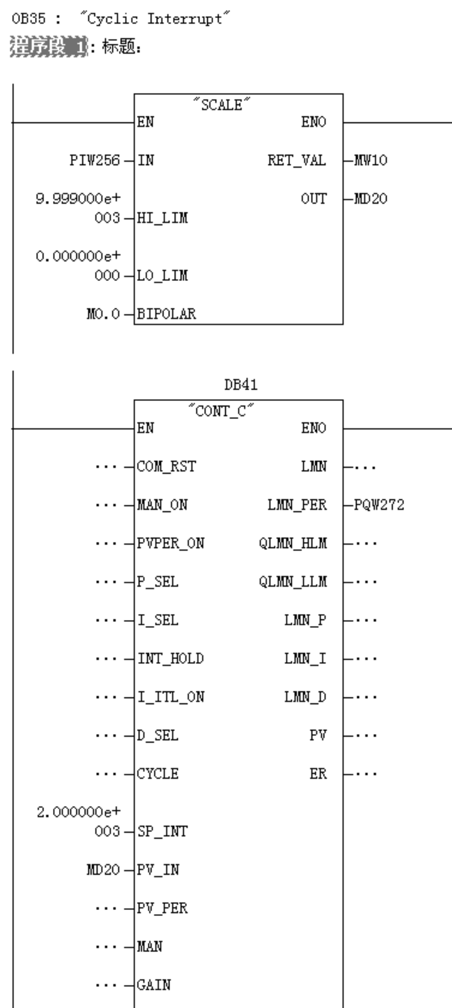


图 9–15　主程序

习题

9-1　推导 PID 模拟化算式离散化数字化算式过程。

9-2　简述 FC105 模块各参数意义及其将数字量规格化工程量过程。

9-3　简述 FB41 实现 PID 运算的过程。

9-3　简述 FB41 与 FB43 结合实现脉宽调制输出过程。

第 10 章　PLC 控制系统设计

PLC 控制系统设计是一个理论问题与工程实际相结合的过程。其设计一般先应根据现场环境状况、工艺特点、控制规模以及控制要求等情况，确定系统总体控制方案、硬件选型、通信网络结构、控制算法。然后进行系统设计，生产现场联调并不断完善，直到满足系统控制性能要求。

10.1　系统设计的原则和内容

10.1.1　设计原则

在实际控制系统中，由于被控对象往往是不一样的，使得控制系统的硬件结构、网络通信方式、控制算法也不一样，但在控制系统的设计及实施过程中所遵循的设计原则是大致相同的，在设计 PLC 控制系统时，一般应遵循以下原则。

（1）可靠性

安全可靠是一个控制系统的基本要求。保证 PLC 控制系统能够长期安全、可靠、稳定的运行是设计系统时必须考虑的。首先需要确保系统即使在恶劣的环境下仍能可靠运行，其次要保证系统能防止误操作。

（2）经济实用性

在满足控制性能要求的前提下，应力求经济实用，不宜盲目追求软硬件系统的高指标、高性能。

（3）可扩展性

在设计实际系统时，应该适当考虑系统的可扩展性，无论是 PLC 容量、I/O 点数还是网络接口，应适当留有一定裕量，以满足因今后生产过程的改进或扩展带来的需求。

（4）易操作性

设计系统时应以人为本，充分考虑操作人员的思维与习惯。设计的人机界面应通俗易懂，便于操作员理解与操作。

10.1.2　设计内容

PLC 控制系统的设计内容主要包括系统总体结构、通信网络方式、硬件系统设计、程序编写、人机界面设计（需要在上位机上操作整个控制系统时）以及技术文档的编写、整理与归档几个方面。

① 根据控制对象实际被控 I/O 点数的多少确定控制系统的总体结构。例如，采用一个 PLC 还是多个 PLC，每个 PLC 带多少个模块等。

② 系统的网络结构是依据系统规模的大小来确定的。例如，如果系统规模比较大，传输的数据量多，可以考虑采用以太网或 DP 网。通信网络方式的设计对系统至关重要，合理的网络结构可以保证数据安全快速的传输。

③ 硬件系统的设计主要包括 PLC 及其各模块的选型、电气控制柜设计以及 I/O 模块端子接线图设计。I/O 模块端子多，在绘制 I/O 端子接线图时，必须严谨细致。

④ 程序设计主要是 PLC 控制程序的编写。依据工艺特点与控制要求，首先设计控制方案，然后依据控制方案进行程序设计。程序是整个控制的核心，程序的好坏直接影响到控制效果，因此，在设计程序时必须考虑有效性与简洁性。

⑤ 当控制系统需要在上位机上操作时，还应设计人机界面。人机界面设计要做到两点，一是内容要能反应工艺流程主体，不需要控制的设备且不是重要的设备可以不体现出来。二是界面要简洁友好，便于操作员操作。

⑥ 技术文档的编写、整理与归档。项目技术文档主要包括系统说明书、系统布置图、电气原理图、硬件明细表、系统安装调试报告、上位机操作说明书以及系统维护手册。

10.1.3 设计步骤

PLC 控制系统的设计虽然与具体被控对象有关，但设计步骤则是基本一致的。设计一个实际的 PLC 控制项目，其过程可分为四个阶段：明确任务阶段、工程设计阶段、联调及投运阶段。

（1） 明确任务阶段

在设计系统前，根据客户提交的控制任务书，深入生产现场进行调研，了解现场环境情况、工艺流程以及控制性能要求。充分与客户方相关人员进行沟通，逐条进行认真分析，对含义不清的，双方协商讨论并最终确认。最后对控制任务书的可行性进行论证，主要包括技术可行性、进度可行性以及费用可行性。若论证可行，则可以与客户签订合同，合同内容包括经双方确认的任务书、付款方式、进度要求、验收方式、违约责任以及其他一些约定。

（2） 工程设计阶段

这部分主要包括硬件设计、程序编写以及人机界面设计。根据控制任务书统计被控点数，确定控制系统的规模，设计系统的总体结构。根据现场环境情况与工艺特点确定系统网络结构。对硬件进行选型，包括 PLC 型号、I/O 模块型号等。分配 I/O 点，设计 I/O 模块端子与输入输出设备的电气接线图。设计控制柜以及控制柜中的原件布置图。根据控制要求以及工艺特点，设计控制方案，画出控制方案图，根据控制方案设计控制程序，并对程序进行仿真调试并不断完善修改直到无误为止。最后设计人机界面，画出工艺流程图、设计报警页面、趋势页面以及报表。

（3） 联调及投运阶段

系统硬件与程序设计完成后，即可将其与现场设备连接进行联调。联调分为两个阶段。一是单机调试，就是只对单个设备进行控制或单个工艺量进行检测。二是对整个生产过程进行调试。只有单机调试无误后方可对整个生产过程进行调试。若在调试过程中存在问题，则要仔细分析找出原因并对硬件或程序做相应修改。

10.2 硬件设计

10.2.1 PLC 的选型

对于 PLC 的选型，在满足控制要求的前提下，注重的是性价比，一般应遵循以下几个

方面。

（1）结构方面

按照物理结构，PLC 分为整体式和模块式。整体式模块所带的 I/O 点数比较少，I/O 模块的种类、输入输出点数的比例以及特殊模块的使用等方面灵活性不够强。模块式在以上几个方面比整体式选择余地多，且维修时更换模块方便。对于稍微复杂点的系统应该选用模块式 PLC。另外，根据安装方式，系统分为集中式、远程 I/O 式和多台 PLC 联网式。这主要取决于系统规模。集中式的 CPU、电源、I/O 模块在一个支架上。远程 I/O 式有多个支架，每个支架上配置有 I/O 模块及电源，支架之间通过通信接口模块相连。联网式有多台 PLC 通过网络连接。集中式适用于规模比较小的系统，远程 I/O 式适用于一般大的系统，联网式 PLC 适用于规模比较大的系统，虽然每台 PLC 独立控制一部分设备，但相互之间又存在联系。

（2）功能方面

当控制系统功能比较简单，例如只是处理些开关量和少量模拟量时，可选用低档 PLC。若控制系统需要 PID 运算、闭环控制、通信联网等功能时，则要选用中高档 PLC。

（3）实时性方面

对于大多数应用场合，PLC 的实时性不是问题。对于一些特殊场合，实时性要求较高，若此时系统规模比较大，点数多，用户程序必然比较长，扫描周期有点长，此时选择 CPU 处理速度快的 PLC，同时可以考虑采用高速响应模块和中断输入模块来提高系统的实时响应性。

（4）联网功能方面

对于大型系统，可能存在不同生产厂家的设备连在一个网络上，互相进行数据通信。因此，要将 PLC 纳入整个工厂自动化控制网络，应选用具有相应通信功能的 PLC。

10.2.2 I/O 模块的选型

PLC 控制系统与生产过程的联系是通过 I/O 接口模块来实现的。通过 I/O 接口模块可以将检测的各种工艺参数输给 PLC 控制器，同时，控制器又可以将运算好的数据通过 I/O 接口模块送给执行机构来实现对设备或生产过程的控制。I/O 接口模块主要包括开关量输入模块、开关量输出模块、模拟量输入模块、模拟量输出模块以及一些特殊模块（例如：称重模块、称重模块）等，根据实际需要进行选用。

1. 开关量输入模块选择

PLC 的开关量输入模块用于检测来自现场（如行程开关、压力开关、按钮）的高电平信号，并将其转换为 PLC 内部的低电平信号。

按输入点数分：常用的有 8 点、12 点、16 点、32 点等开关量输入模块。

按工作电压分：常用的有直流 5 V、12 V、24 V，交流 110 V、220 V 等开关量输入模块。

选择开关量输入模块主要考虑以下两点。

① 根据现场信号（如开关、按钮）与 PLC 输入模块距离的远近来选择电压高低。一般距离较近的设备选用较低电压模块，距离较远的设备选用较高电压模块，如 12 V 电压模块一般不超过 10 m。

② 点数多的模块，如 32 点开关量输入模块，允许同时接通的点数取决于输入电压和环

境温度。一般同时接通的点数不宜超过总输入点数的60%。

2. 开关量输出模块选择

开关量输出模块的功能是将PLC内部低电平的控制信号转换为外部所需电平的输出信号，驱动外部负载。开关量输出模块有三种输出方式：继电器输出、双向晶闸管输出以及晶体管输出。

（1）输出方式的选择

继电器输出方式价格便宜，承受瞬时过电压和过电流能力较强，且具有隔离作用。但继电器有易触电，寿命短，且响应速度较慢，适用于动作不频繁的交直流负载。当驱动电感性负载时，最大开关频率不得超过1 Hz。双向晶闸管输出（交流）和晶体管输出（直流）都属于无触点开关输出，适用于通断频繁的感性负载。

（2）输出电流的选择

模块的输出电流必须大于负载电流的额定值，同时，考虑到接通时的冲击电流效应以及干扰等因素，还要考虑留有一定的容量。若负载电流较大，则输出模块不能直接驱动负载，应该增加中间环节（如通过一个型继电器来实现）。

（3）驱动能力

在选用输出模块时，不但要看一个点的驱动能力，还要看整个输出模块满负载驱动能力，同时接通时的总电流值不能超过模块允许的最大电流。

3. 模拟量I/O模块选择

模拟量I/O模块是用于接收传感器产生的信号和输出模拟量控制信号。这些接口能测量压力、流量、温度等模拟量的数值，并输出电压或电流模拟信号驱动设备。典型的模拟信号类型有－10 V～10 V，0 V～10 V，4 mA～4 mA等。

4. 特殊功能I/O模块选择

在实际工程中，可能要用到一些特殊模块，如位置控制模块、计数器模块、称重模块以及闭环控制模块等，这些模块再根据特定需要选用。

10.2.3 PLC容量估算

PLC容量包括I/O点数和用户存储器的存储容量两个方面。在做实际工程系统时根据系统规模合理选择，在满足控制系统要求的情况下适当留有裕量以便备用。

（1）I/O点数的确定

PLC的I/O点数就是被控对象的实际的输入输出点数，在设计控制系统时，一般选择系统实际点数的110%～115%作为PLC的I/O点数。

（2）用户存储器容量的确定

用户程序占用多少内存与许多因素有关，如I/O点数、程序结构、运算处理量等。在计算存储器容量时，常采用估算法。根据经验，每个I/O点及有关功能器件占用的内存如下。

开关量输入：所需存储器字节数＝输入点数×10。

开关量输出：所需存储器字节数＝输出点数×8。

模拟量输入：所需存储器字节数＝通道数×100。

模拟量输出：所需存储器字节数＝通道数×200。

定时器/计数器：所需存储器字节数＝定时器/计数器×2。

通信接口：所需存储器字节数 = 接口个数 ×300。

在实际选择 PLC 的容量时，按存储器总字节数的 110% ~125% 作为所需存储器的容量，一般计算公式为

所需存储器容量(KB) = (1.1 ~1.25) × (DI ×10 + DO ×8 + AI ×100 + AO ×200 + T/C ×2 + CP ×300)/1024

其中 DI 为开关量输入总点数；DO 为开关量输出总点数；AI 为模拟量输入通道总数；AO 为模拟量输出通道总数；T/C 为定时器/计数器总数；CP 为通信接口总数。

10.3 软件设计

10.3.1 设计前准备工作

（1）熟悉编程软件

编程软件是程序设计的前提。在设计程序前，应该通过在计算机上实际操作来熟悉编程软件的结构及使用方法，熟悉各种指令的含义。

（2）设计程序框图

程序框图是程序思想的体现，是编程的依据。在程序设计前，应根据控制要求确定用户程序结构以及详细的程序框图。程序框图应尽量做到模块化，确定模块的输入及输出，确定模块完成的功能。同时弄清各模块之间的联系。

（3）变量表的定义

变量包括三部分，一部分是 I/O 模块各通道对应的信号变量，一部分是需在上位机上显示或操作的变量，一部分是在各程序块中设定的局部中间变量。其中第三部分在设计程序块中定义。前两部分应该列表详细定义各变量的意义，包括变量名称、变量地址、变量对应的工艺参数名称。变量的命名应该遵循一定的规则，尽量做到能从名称中可以知道其所代表的设备名或工艺参数。

10.3.2 编写程序

在完成了设计程序前的准备工作之后就可以根据程序框图编写程序了。编写程序时，尽量采用模块化与结构化设计。同时多借鉴一些典型的程序，如单电机起动程序、双电机起动程序、单回路 PID 控制程序、多设备顺控程序、给料调节程序等。另外，要对程序做好注释，增强程序的可读性。这点非常重要，主要体现在两个方面：首先，一个大型项目程序往往是多人共同来编写，相互之间需要交流，增加注释有利于交流。其次，有利于维护人员后续对系统的维护。

10.3.3 程序测试

刚编好的程序不可避免地存在错误与缺陷，因此，程序编写完后，要对程序进行测试，检验程序的正确性，确保程序能实现应有的功能。程序经排错、修改、测试无误后方可下载到 PLC 的 CPU 中与现场设备联调。

10.4 系统调试

10.4.1 调试步骤

系统调试时，应首先按要求将电源、I/O 端子、网络接口等外部接线连接好，确保系统与现场设备连接无误，然后将已经编写好的程序下载到 PLC 使其处于监控或运行状态，开始调试。系统调试流程如图 10-1 所示。

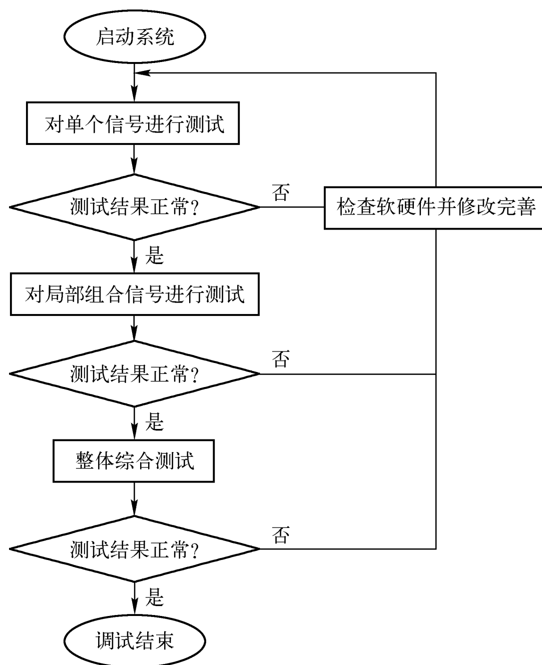


图 10-1　系统调试流程图

10.4.2 调试方法

（1）对单个信号进行调试

控制系统的信号是由一个个单独的信号组成的。在对系统进行调试时，先要对单个信号进行测试。例如，测试电机起动时，给一个起动指令，观察 PLC 对应开关量输出端子指示灯是否亮，同时观察设备是否已起动。如果在测试过程中发现不符合要求的地方，先检查接线是否正确，当接线无误时再检查程序并加以修改完善。直到每个单个信号测试符合系统性能要求为止。

（2）对局部组合信号进行调试

因为在工艺上联系紧密，系统中有些信号存在联锁关系。在调试时，将这部分信号视为局部组合信号。对这部分信号进行调试主要测试各程序相互之间的关系是否满足要求。例如，几台设备的起、停存在顺序之分且互为条件时，就应将这几台设备的起停程序作为局部组合程序来测试，发出起停信号后，观察设备是否按要求运行。若出现问题，应检查程序并

加以修改，直到满足系统功能要求为止。

（3）整体综合调试

整体综合测试是对整个现场信号进行模拟实际运行情况进行测量和控制，观察整个系统的运行状态和性能是否符合控制性能要求。若在此过程中，达不到控制性能要求，应从软硬件两方面加以分析，找出解决办法，完善系统使控制达到要求。

10.5 应用实例

10.5.1 水箱水温自动控制系统

在工业生产中，温度是常见的一个过程变量。例如：在冶金工业、化工工业、机械加工和食品加工等许多领域，都需要对各种加热炉、热处理炉、反应炉和锅炉的温度进行控制。本节以水箱水温控制为例来说明 PLC 控制系统的设计过程。

10.5.2 功能需求

本水箱水温控制系统，要求对水温加热至设定温度。PLC 温度控制系统的具体指标要求是：超调量≤±1℃，稳态误差≤±0.5℃。在上位机实现温度曲线监控。

10.5.3 控制方案设计

根据水箱水温控制功能需求，控制方案设计如下：利用电阻加热丝作为加热元件，将可控硅作为执行元件，接入到 220 V 加热主电路当中。选用西门子 S7－300 PLC 作为控制器、热电阻 Pt100 作为检测元件、AI－808 仪表作为变送单元组成一个闭环系统。传感器将检测到的温度值送入 PLC 中与设定温度进行比较，差值经 PID 运算后输出脉冲信号，该信号控制可控硅的通断来实现加热过程。用一台计算机作为上位机，利用力控组态软件设计人机界面以及温度实时曲线图，上位机与 PLC 之间通过 MPI 进行通信。水箱水温控制系统结构框图如图 10-2 所示。

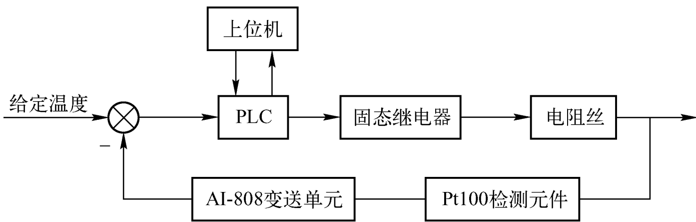


图 10-2　系统结构框图

10.5.4 硬件系统设计

1. 硬件选型

（1）点数统计

本系统比较简单，只有一个模拟量输入点，一个数字量输出点，统计见表 10-1。

表 10–1　点数统计

序　　号	信 号 类 型	点　　数
1	模拟量输入	1 路（1 ~ 5 V）
2	数字量输出	1 点

（2）硬件选型

根据系统控制方案要求以及点数统计情况，硬件选型如下：PLC 控制器选用 CPU315 – 2DP；模拟量输入模块选用 SM334，含 4 路模拟量输入，2 路模拟量输出；数字量模块选用 SM322，16 路数字量输出；固态继电器选用额定电流为 10 A、额定电压为 250 V 的；电源模块选用电流为 2 A 的；温度传感器选用 Pt100；变送单元选用 AI – 808 仪表；普通计算机一台；MPI 编程电缆一根。系统硬件设备表见表 10–2。

表 10–2　系统硬件设备选型

序　　号	名　　称	型　　号	订　货　号	数量	单位
1	安装导轨	530 mm	6ES7390 – 1AF30 – 0AA0	块	1
2	电源模块	PS307 2A	6ES7307 – 1BA00 – 0AA0	块	1
3	CPU	CPU315 – 2DP	6ES7315 – 2AH14 – 0AB0	块	1
4	模拟量模块	SM334（AI4/AO2）	6ES7334 – 0CE01 – 0AA0	块	1
5	数字量输出模块	SM322（16 点/DC24V）	6ES7322 – 1BH01 – 0AA0	块	1
6	编程电缆	USB　PC Adapter	6ES7 972 – 0CB20 – 0XA0	根	1
7	固态继电器	SSR – 10 DA		个	1
8	变送单元	AI – 808		块	1
9	温度检测元件	Pt100（ – 200℃ ~ 420℃）		套	1

2. 电气接线与参数设置

系统的电气连接如图 10–3 所示。热电阻接入电路是一个不平衡电桥，热电阻作为电桥的一个桥臂，其连接导线也是桥臂的一部分，在计算温度时，其连接导线电阻当作了热电阻

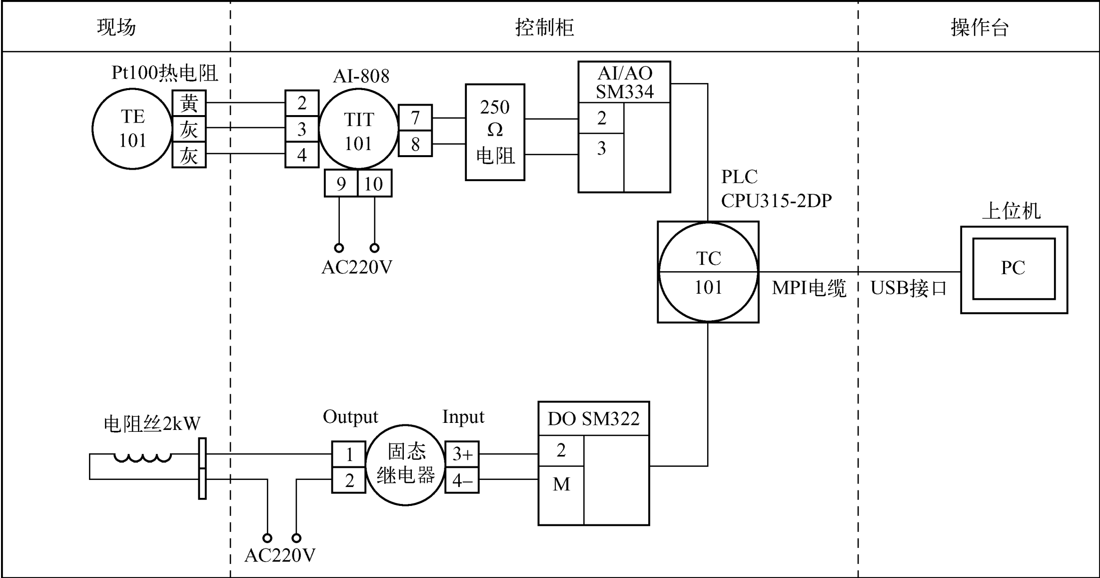


图 10–3　系统硬件接线图

的一部分，因而会引起测量误差。所以本设计中热电阻采用三线制，目的在于尽量消除连接导线电阻所引起的测量误差。热电阻的黄线与两根灰线分别接到 AI－808 仪表的 2、3、4 端子，经 7、8 变送输出 4～20 mA 的电流信号，并接 250 Ω 电阻后转换为 1～5 V 的电压信号接到 SM334 的 2、3 端子。检测到的温度信号送入 PLC 控制器中与来自上位机设定的温度信号进行比较并进行 PID 运算，根据运算结果，PLC 经 SM322 模块 2、M 端子输出脉冲信号控制固态继电器的通断，从而控制电阻丝的通电占空比，以此来实现水箱温度的控制。

AI－808 仪表需要进行参数设置，具体情况如下。

Sn（输入信号规格）：21；

Ctrl（控制方式采用为调节）：0；

DIL（输入下限显示值）：0；

DIH（输入上限显示值）：100；

OP1（选择 4～20 mA 线性电流输出）：4；

DIP（显示值小数点位数：）：1。

10.5.5　程序设计

1. 设计方案

本控制系统中，由于对电加热丝的控制采用固态继电器作为执行器，所以程序设计采用西门子 PLC 自带的 FB41 软件模块与 FB43 软件模块相结合的方法来实现。FB41 的模拟量输出信号经 FB43 调制成占空比可调的脉冲信号来控制加热丝的通断。

（1）FB41 的算法原理

FB41“CONT_C”是采用位置式 PID 算法思想设计的控制软件模块。FB41 的算法设计很完善，使用起来也很灵活。它的比例运算、积分运算（INT）和微商运算（DIF）是并行连接的，可以单独激活或取消。这就允许组态成 P、PI、PD 和 PID 控制器。它的积分分量可以清零、保持，这就方便实现抗积分饱和或积分分离。微分分量既可以直接输出，也可以延迟衰减输出。FB41 输出的是模拟量控制信号。

FB41. CYCLE 参数是 PID 控制的采样周期，也即 PID 的控制周期。即每一个控制周期，PID 算法采样一次过程变量当前值，和设定值比较后，进行 PID 运算，输出控制值给执行器，产生相应的动作，完成一次控制过程。其算法如下：

$$U(k) = Kc\left\{e(k) + \frac{T_s}{T_i}\sum_{i=0}^{k} e(i) + \frac{T_d}{T_s}[e(k) - e(k-1)]\right\}$$

为了保证准确有序的控制，PID 运算应放在 OB35 定时中断服务程序中，放在 OB1 主循环中是不合适的，因为主循环的扫描周期是不固定的，一般和 FB41. CYCLE 设置值不一致且相差较大时，放在 OB1 中将使 CYCLE 设置失去意义。

（2）FB43 的算法原理

FB43“PULSEGEN”称为脉冲宽度调制器，即 PWM 模块。可以将输入变量“INV”（Input Value＝PID 控制器的调节输出 LMN）转换为一个恒定周期的脉冲串，每个周期的脉冲宽度与输入变量成正比。

① FB43. PER－TM：PERIOD TIME（周期时间）脉冲宽度调制的恒定周期。

② FB43. CYCLE：SAMPLING TIME（采样时间）是 FB43 模块的采样时间，也即 PWM

模块的调用周期。而 FB43. PER – TM 周期时间是若干个 FB43. CYCLE 采样时间之和。

③ FB43 模块中定时的实现是靠对时基信号的计数实现的。当把 FB43 模块放在 OB35 定时中断服务程序中时，时基信号就是 OB35 的定时中断时间，也就是 FB43. CYCLE 时间。如果定时中断时间是 100 ms，为了实现 PER – TM = 5 s = 5000 ms 的定时，FB43 模块内部应设置计数器 cnt1，计数器的初值应设置为 5000 ms/100 ms = 50。每个 OB35 周期调用一次 FB43 模块，对 cnt1 减一计数，直至为 0，本 PER – TM = 5 s 周期结束，重装计数初值，开始下一个周期计数。所以 cnt1 是循环计数工作的。

为了实现输出脉宽的控制，FB43 模块内部存在计数器 cnt2。例如当 INV = 30 时，输出脉冲宽度 = (30/100) ×5000 ms = 1500 ms，cnt2 的计数初值设为 1500 ms/100 ms = 15，每个 OB35 周期调用一次 FB43 模块，对 cnt2 减一计数，直至为 0，即实现 QPOS 为 ON 的输出脉宽控制。当 cnt2 计数值减为 0 时，QPOS 端由 ON 状态转为 OFF 状态，直至本 PER – TM 周期结束。下一 PER – TM 周期，根据新的 INV 值，计算 cnt2 新的计数值，开始下一周期的控制。cnt2 也是循环计数的，只是计数初值在每个 FB43. PER – TM 周期是可变的。

（3） FB41 与 FB43 的结合

PID 调节的输出即 FB41. LMN 连接到 FB43. INV 端，经过 FB43 的脉宽调制，在 FB43. QPOS 输出端上将以 SAMPLING TIME 的步长转换成脉冲宽度。脉冲的宽度正比于 INV 的大小。PWM 脉冲发生器的采样时间和“CONT_C”控制器的采样时间之比决定了脉冲宽度调制的精度。

① FB43. PER – TM：当 FB41 和 FB43 联用时，这相当于“CONT_C”控制器的控制周期，应该和 FB41. cycle 设置一样。

② FB43. CYCLE：FB43 模块的调用时间必须恒定，所以应该放在 OB35 定时中断中，并且 OB35 的中断周期设置应该和 FB43. CYCLE 一致。

③ FB41 和 FB43 之间的时序配合。当把 FB41 和 FB43 都放在 OB35 中的时候，就产生了这样一个问题：由于 FB41. cycle 和 FB43. cycle 不一致，每次进入 OB35 中断服务程序 FB43 都应该执行一次，而 FB41 则不然。进入 OB35 中断服务程序（FB41. cycle/FB43. cycle）次，才执行一次 FB41。这样就需要在 OB35 中断服务程序中由用户自主设置一个计数器，设为 counter，初值设为（FB41. cycle/FB43. cycle），每次进入 OB35，counter 减一计数，减为 0 时，执行一次 FB41。然后重装计数初值，开始下一个 PID 循环。FB41 运算后的输出与 FB43 的输出关系如图 10-4 所示。

本控制系统中，设置 FB41. cycle = 3000 ms，FB43. cycle = 100 ms。脉冲调节的精度即是 100 ms，而 100 ms/3000 ms 即是脉冲调节的分辨率。

（4） 程序流程图

根据前面分析的 FB41 与 FB43 相结合的算法原理，整个程序可以设计为三部分，初始化程序、主程序以及中断程序。程序流程图如图 10-5、图 10-6 和图 10-7 所示。

2. 程序编写

（1） 变量表

实际的 PLC 控制系统，一般都是通过上位机对生产过程或设备进行监控与操作。本系统也配备了一台上位机。在编写程序前，需统计 PLC 端子硬接点变量以及上位机人机界面软变量，以便明确其这些变量的地址以及它们所代表信号的物理意义。本系统的相关变量表见表 10-3。

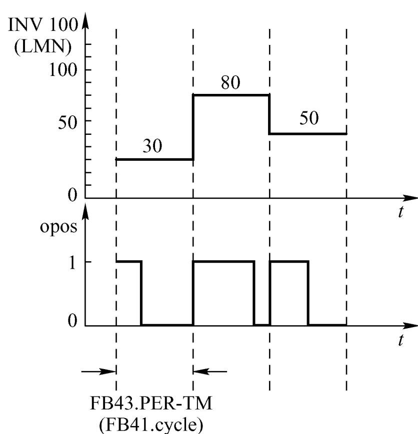


图 10-4　FB41 输出与 FB43 输出关系

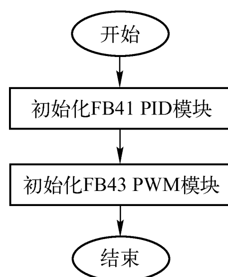


图 10-5　初始化程序流程图

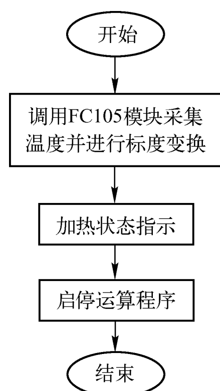


图 10-6　主程序流程图

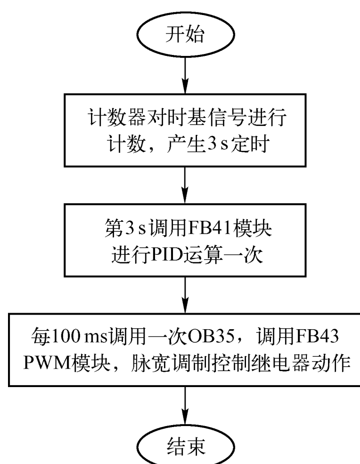


图 10-7　中断程序流程图

表 10-3　PLC 硬变量与上位机软变量表

序　　号	名　　称	变 量 符 号	地　　址	数 据 类 型	注　　释
1	启动按钮	START. PV	M2. 0	BOOL	运算程序启动
2	关闭按钮	STOP. PV	M3. 0	BOOL	运算程序关闭
3	继电器状态	OUT. PV	M10. 0	BOOL	继电器状态
4	数字量输出	Q4. 0	Q4. 0	BOOL	数字量输出
5	模拟量输入	PIW256	PIW256	INT	模拟量输入
6	比例系数	GAIN. PV	DB41. DBX16. 0	REAL	P 参数
7	积分系数	TI. PV	DB41. DBX20. 0	REAL	I 参数
8	微分系数	TD. PV	DB41. DBX24. 0	REAL	D 参数
9	温度设定值	SP. PV	DB41. DBX6. 0	REAL	设定值
10	温度实际值	PV. PV	DB41. DBX10. 0	REAL	设定值

（2）程序

在 OB100 组织块中编写初始化程序，对 FB41 模块与 FB43 模块初始化，程序如图 10-8 所示。

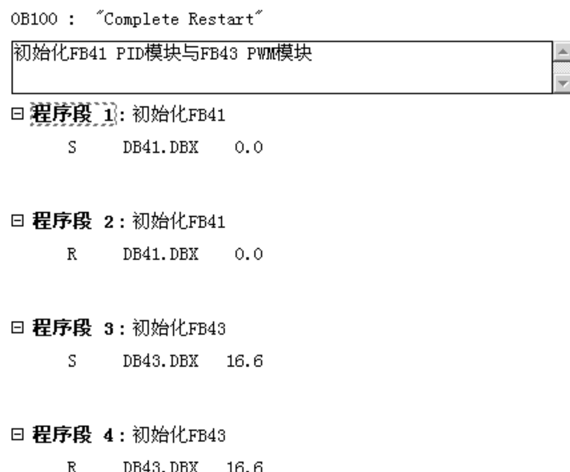


图 10-8 初始化程序

在 OB35 中编写中断程序，OB35 的中断周期为 100 ms。通过设计计数程序实现每隔 3 s 调用一次 FB41 模块。FB43 的采样周期与 OB35 的中断周期一致设为 100 ms。FB43 的脉冲宽度调制周期与 FB41 的采样周期一致设为 3 s。每隔 100 ms 调用一次 FB43 模块。根据 FB41 PID 模块运算结果 LMN，在 3s 的调制周期内 FB43 输出宽度为（LMN/100）×3000 ms 的高电平信号，程序分别如图 10-9、图 10-10 和图 10-11 所示。

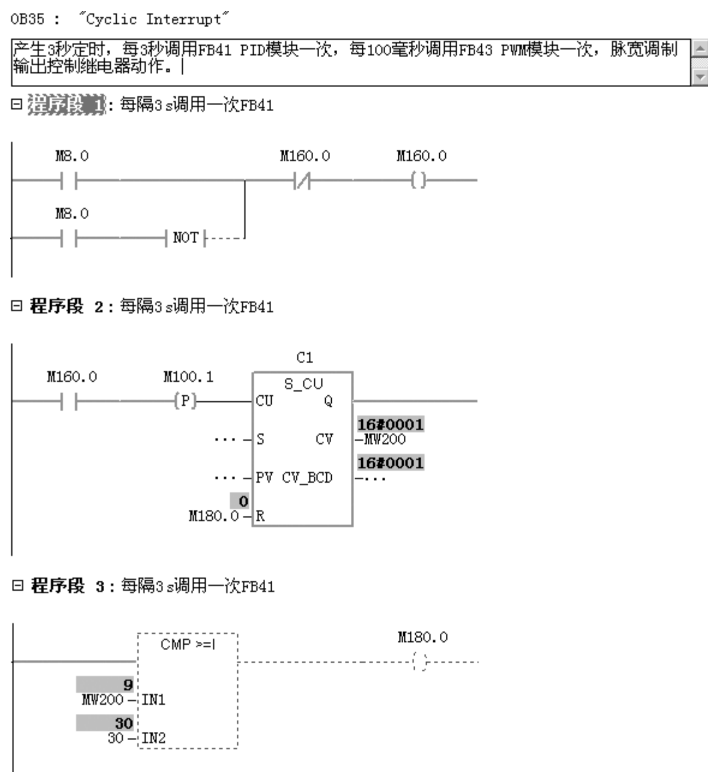


图 10-9 3 s 周期信号产生程序

图 10-10　PID 运算程序

图 10-11　脉宽调制程序

主程序主要是完成温度数据的采集与标度变换，继电器开断状态显示以及 PID 运算程序启停控制，程序如图 10-12 所示。

10.5.6　上位机软件设计

上位机软件可以实现系统的人机界面设计、报警页面设计、趋势页面设计以及报表设计等。由于本系统简单，不涉及报警与报表，在力控组态软件平台下只设计了人机界面以及趋势页面。可通过上位机对系统进行监控，包括设定温度目标值、PID 运算模块的比例系数、积分系数、微分系数、运算程序的启停，显示温度当前值、温度的实时趋势以及继电器通断状态也即加热丝通电状态。

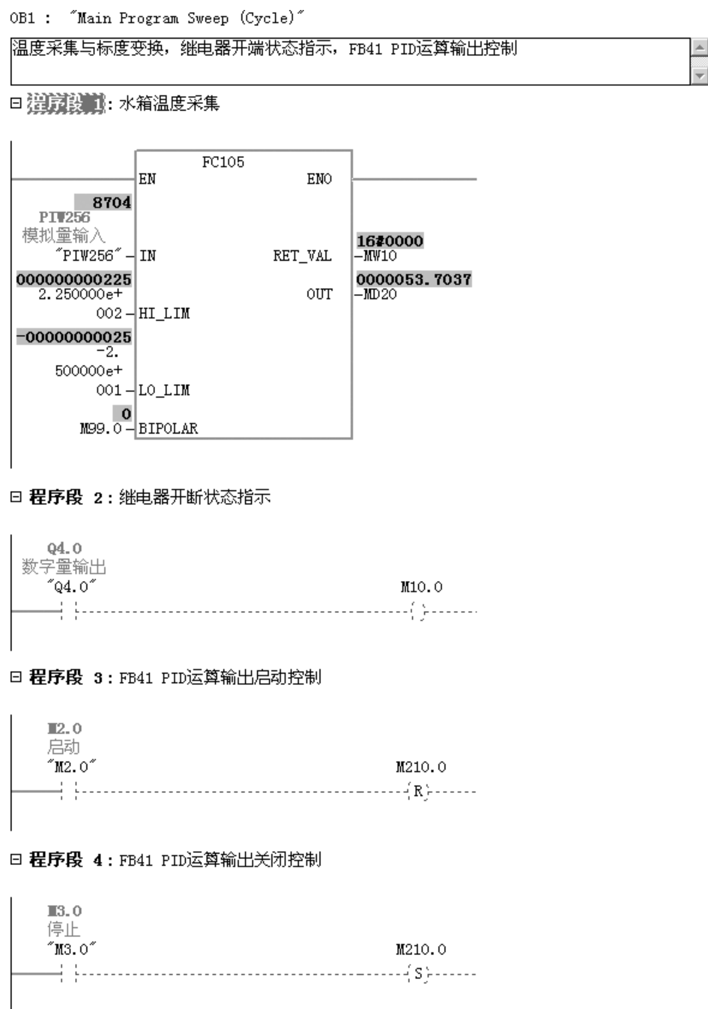


图 10-12　主程序

（1）人机界面设计

人机界面设计要将系统主要流程界面显现出来，布局要合理，所有的显示或操作变量要体现出来，便于操作员对系统进行监控。本系统简单，将上位机、PLC、电源开关以及继电器都绘制出来了，实际项目中，这些都不需要绘制。人机界面设计如图 10-13 所示。

（2）趋势页面设计

趋势页面设计的目的在于监控系统中模拟量，通过曲线显示不同时间各个模拟量的值。不同模拟量应以不同颜色的曲线表示，并要在页面中加以标注，便于区别。本系统中只有一个温度量。温度监控页面设计如图 10-14 所示。

10.5.7　系统调试

当 PLC 程序设计完成且仿真调试无误以及上位机监控软件设计完成后，需要将系统与现场设备连接进行联调。首先，对单个信号进行调试，本例中，先调试温度采集信号，检查上位机是否能采集到水箱水温，无误后，再人为将 Q4.0 置 1，检查继电器是否得电导通，

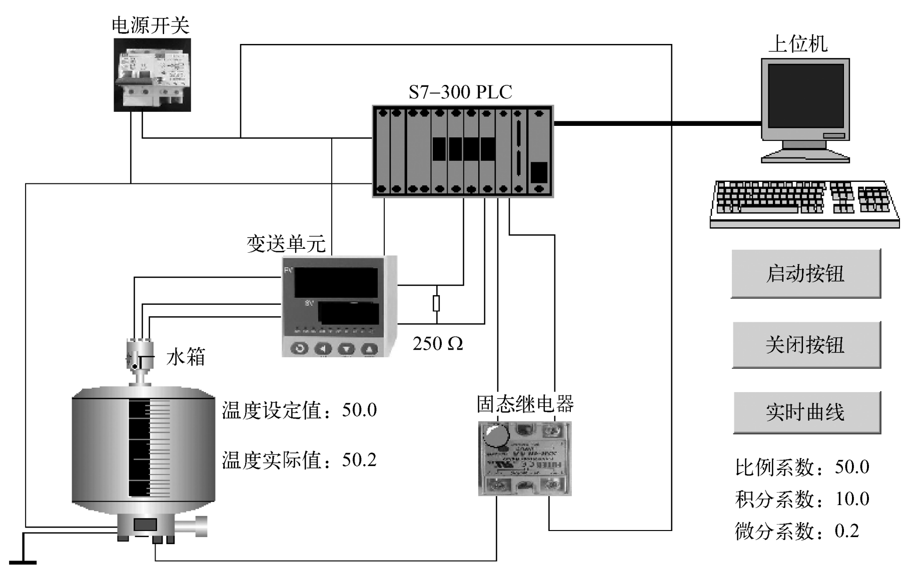


图 10-13　系统人机界面

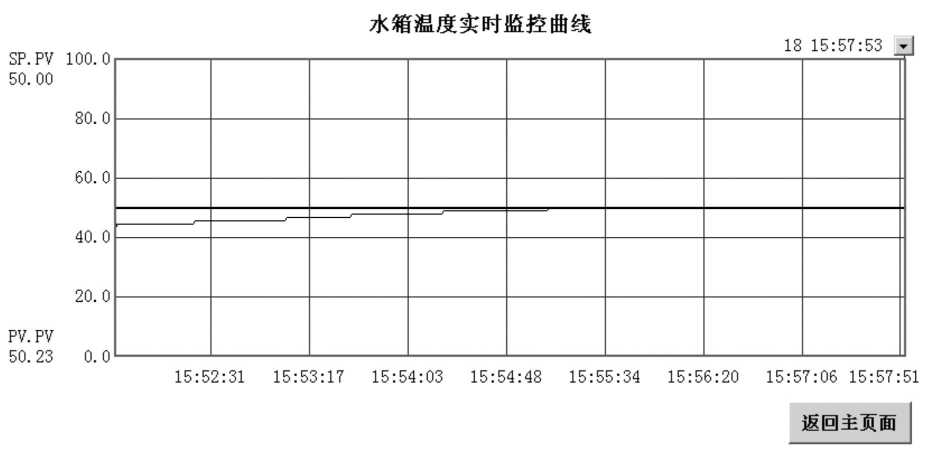


图 10-14　温度监控曲线

系统开始加热。然后进行整体调试，若出现问题，应从软硬件两方面仔细分析并加以完善，直到符合系统控制要求为止。

习题

10-1　简述 PLC 控制系统设计的原则。

10-2　简述 PLC 控制系统的设计步骤。

10-3　简述 PLC 控制系统硬件选型注意事项。

10-4　简述 PLC 控制系统调试方法。

参 考 文 献

[1] 刘忠超．西门子 S7－300 PLC 编程入门及工程实践[M]．北京：化学工业出版社，2015.
[2] 刘忠超．电气控制与可编程自动化控制器应用技术——GE PAC[M]．西安：西安电子科技大学出版社，2016.
[3] 刘忠超．组态软件实用技术教程[M]．西安：西安电子科技大学出版社，2016.
[4] 訾鸿，赵岩，周宝国．S7－300/400 系列 PLC 入门及应用实例[M]．北京：电子工业出版社，2012.
[5] 廖常初．跟我动手学 S7－300/400 PLC[M]．北京：机械工业出版社，2010.
[6] 阳胜峰．S7－300/400 PLC 技术视频学习教程[M]．北京：机械工业出版社，2012.
[7] 柳春生．电器控制与 PLC（西门子 S7－300 机型）[M]．北京：机械工业出版社，2015.
[8] 夏田，陈婵娟，祁广利．PLC 电气控制技术——CPM1A 系列和 S7－200[M]．北京：化学工业出版社，2010.
[9] 刘华波，刘丹，赵岩岭，马艳．西门子 S7－1200 PLC 编程与应用[M]．北京：机械工业出版社，2011.
[10] 陈海霞．西门子 S7－300/400 PLC 编程技术及工程应用[M]．北京：机械工业出版社，2013.
[11] 姜建芳．电气控制与 S7－300 PLC 工程应用技术[M]．北京：机械工业出版社，2014.

在线互动交流平台

官 方 微 博：http://weibo.com/cmpjsj
读者教师QQ群：427066270
33811098

电气控制与西门子S7–300 PLC编程技术


普通高等教育电气信息类规划教材

- C语言程序设计 蔡启先
- 微机原理及接口技术 蔡启仲
- 单片机原理及应用 韩骏峰
- 单片机应用技术 陈桂友
- 单片机原理与应用 杭和平
- 单片机原理及应用 王景景
- 单片机原理及应用 蔡启仲
- 电磁场与电磁波 阳小明
- 电机与拖动基础 徐胜军
- 电力电子技术 第3版 周渊深
- 机电传动控制 张志义
- 电子工艺实训基础 孙 蓓
- 电气控制与西门子S7–300 PLC编程技术 刘忠超
- 机电一体化技术 郭文松
- 数据采集技术与系统设计 李念强
- 自动化专业英语教程 王彩霞
- 信号与控制基础 张家海
- CPLD/FPGA控制系统设计 周京华
- PLC基础及应用教程（三菱FX2N系列） 秦春斌
- 西门子S7–200 PLC应用教程 赵全利
- 传感器技术实用教程 吕勇军
- EDA技术与应用 孙志雄

地址：北京市百万庄大街22号
邮政编码：100037
电话服务
服务咨询热线：010-88379833
读者购书热线：010-88379649
网络服务
机 工 官 网：www.cmpbook.com
机 工 官 博：weibo.com/cmp1952
金 书 网：www.golden-book.com
教育服务网：www.cmpedu.com
封面无防伪标均为盗版


机工教育微信服务号


IT有得聊微信服务号


机工社本科电类教师群
扫一扫二维码，加入该群

上架指导 自动化/PLC

ISBN 978-7-111-57621-1

策划编辑◎尚 晨 / 封面设计◎





定价：49.80元